AMENDMENTS TO THE DRAWINGS

The attached sheets of annotated drawings specifies the changes to Figures 2-4 and 31. In Figure 2, the previously unidentified Additional Sensors has been identified with Reference Number 200C. Figure 3 now identifies the previously unidentified I/O and Digital Processing Circuitry. Figure 4 now identifies previously unidentified Sinc1 and Sinc2 and Figure 31 more correctly identifies the Data Path. All changes have been highlighted for the Examiners convenience.

Applicants concurrently submit formal drawings directly to Mail Stop PG PUB DRAWINGS to replace the existing drawings for this application. Applicants believe these changes will put the drawings in conformance with the specification. Attached is a copy of the submission of the formal drawings for the examiner's convenience.

No new matter has been added, Figures 2, 3, 4 and 31 have been merely amended to more particularly point out and distinctly illustrate the subject matter Applicants believe is inventive.

Attachment: Replacement Sheets
Annotated Sheets Showing Changes

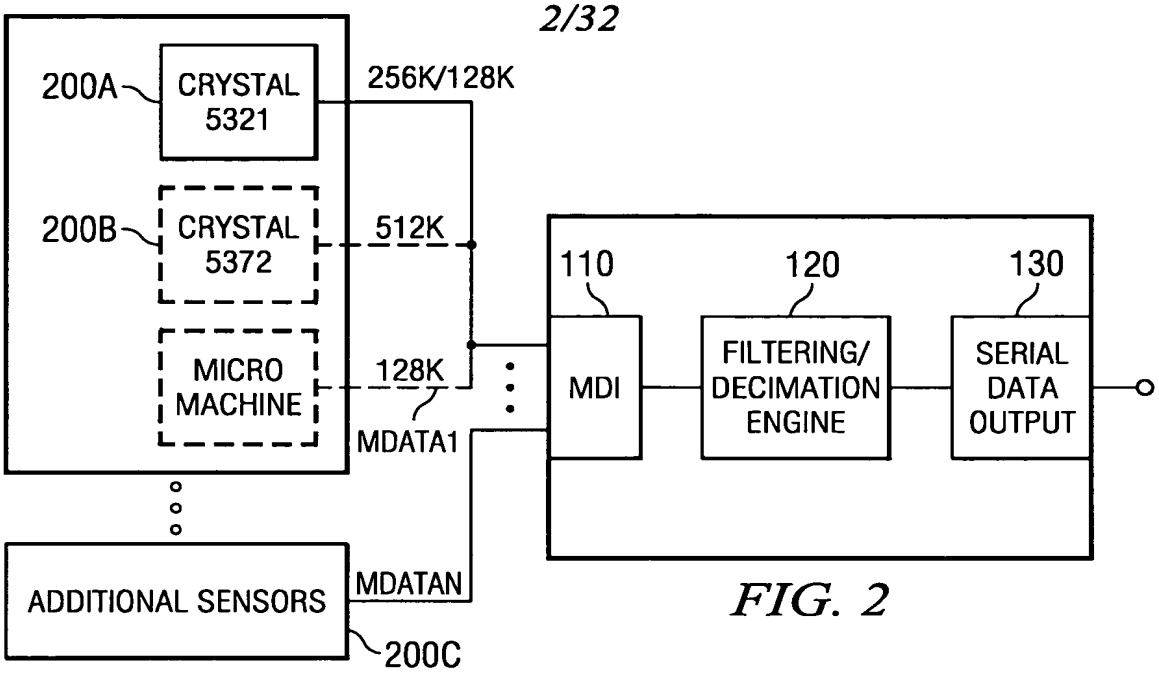


FIG. 2

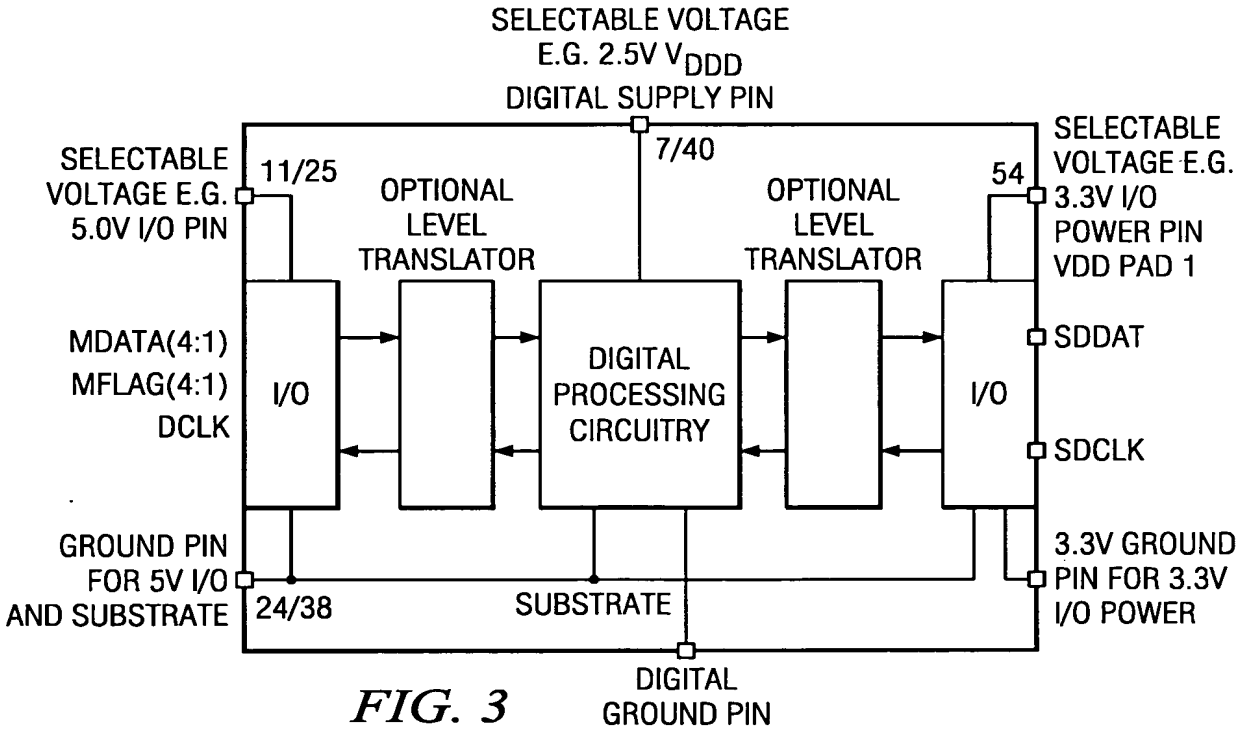


FIG. 3

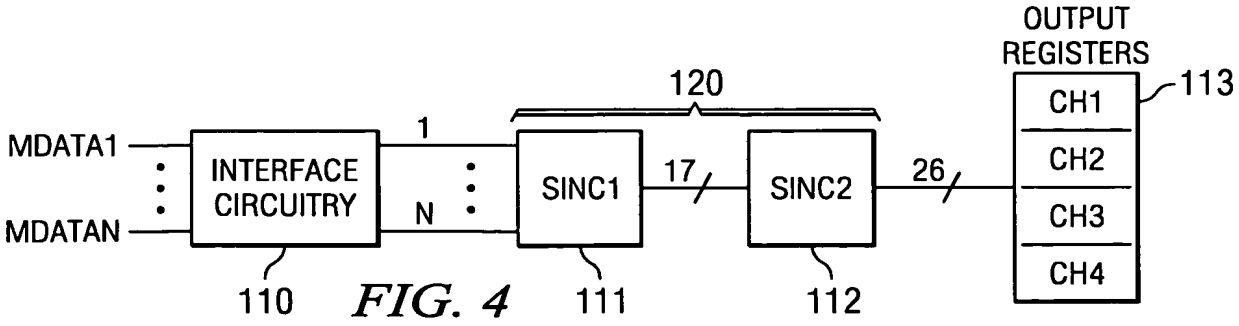


FIG. 4

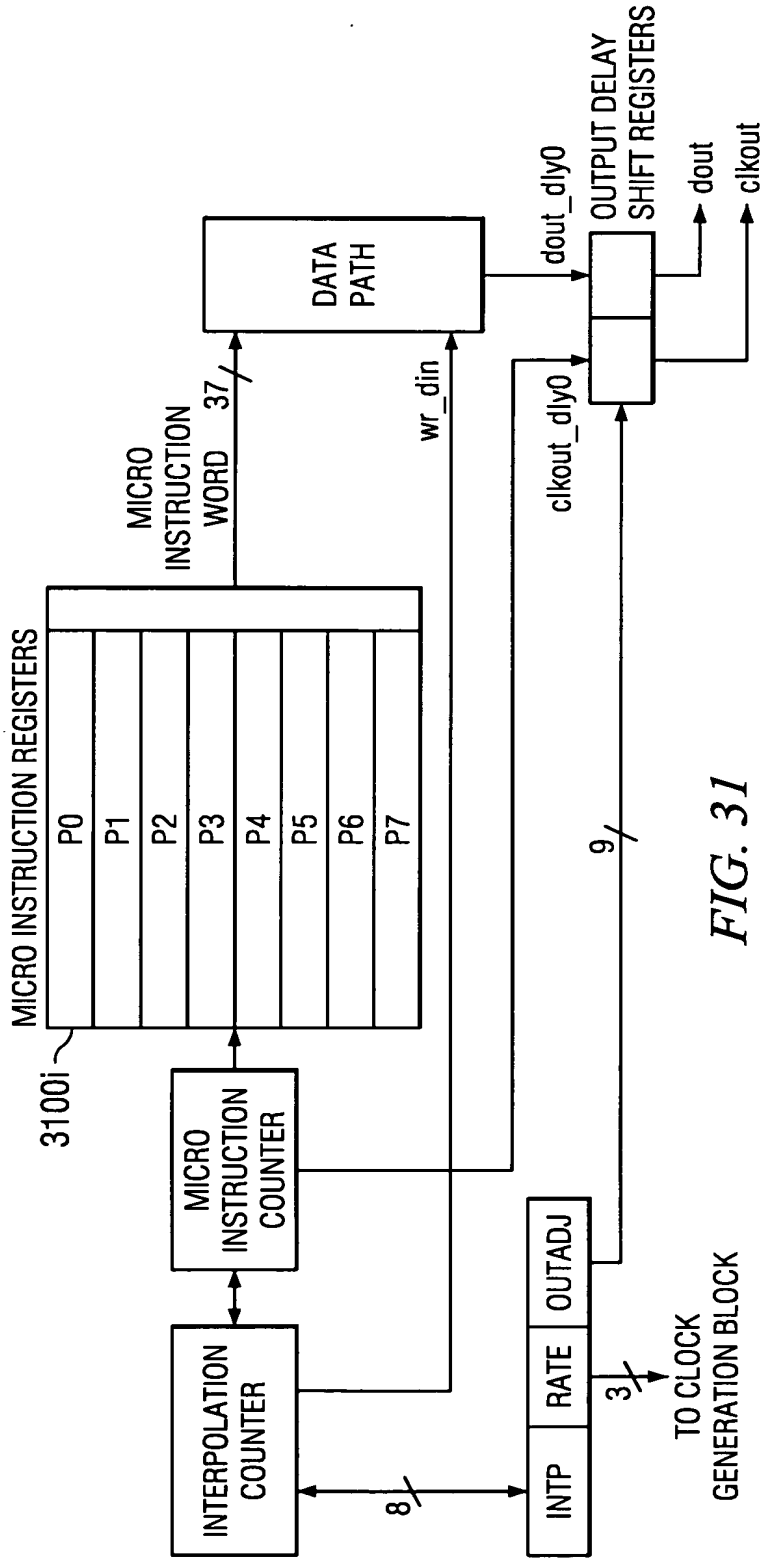


FIG. 31

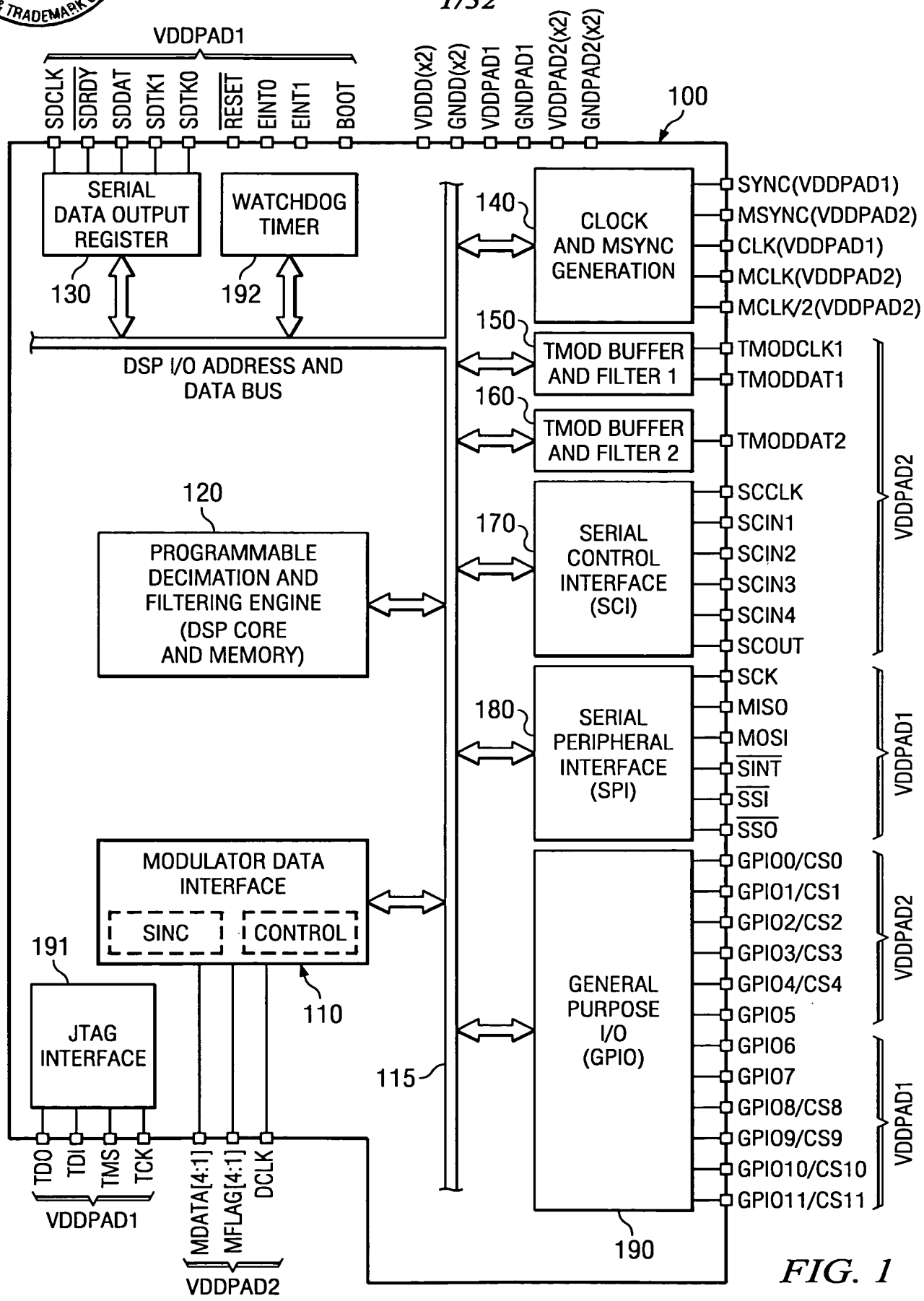


FIG. 1

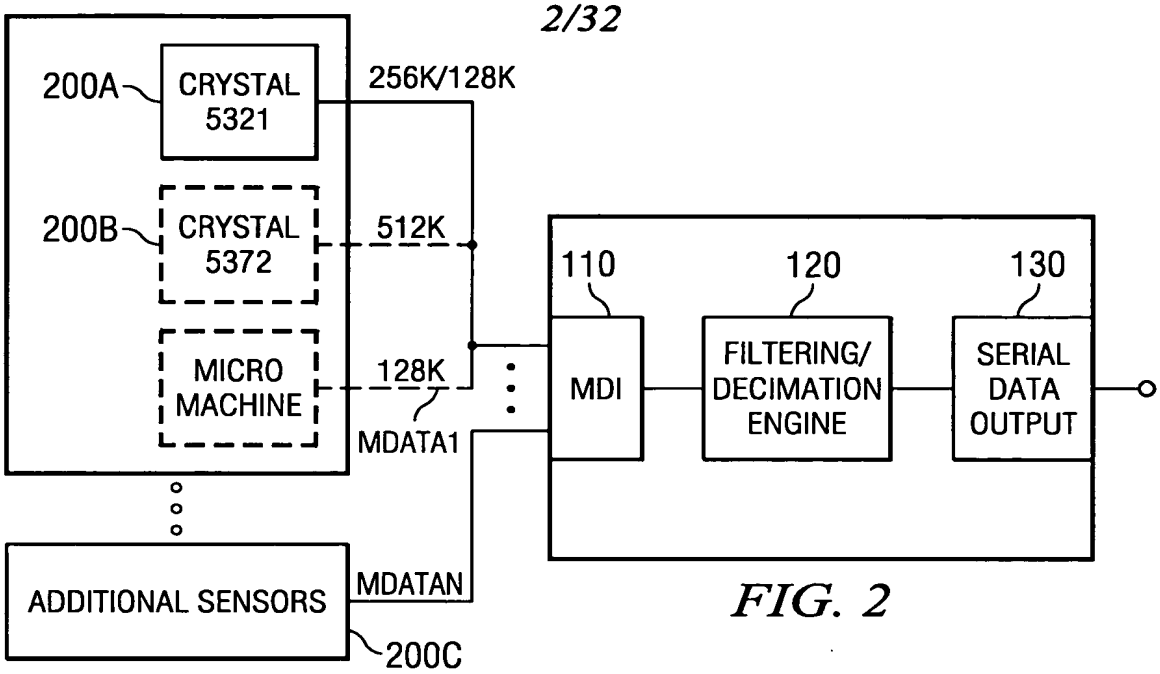


FIG. 2

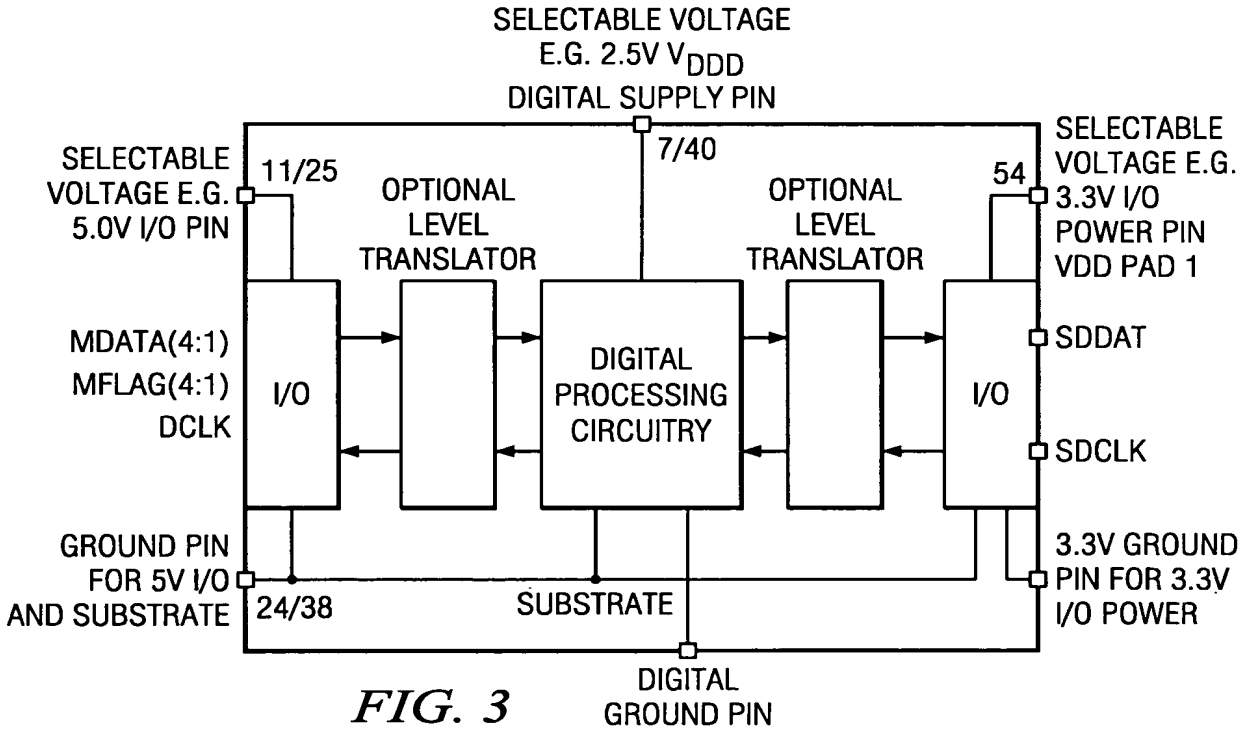


FIG. 3

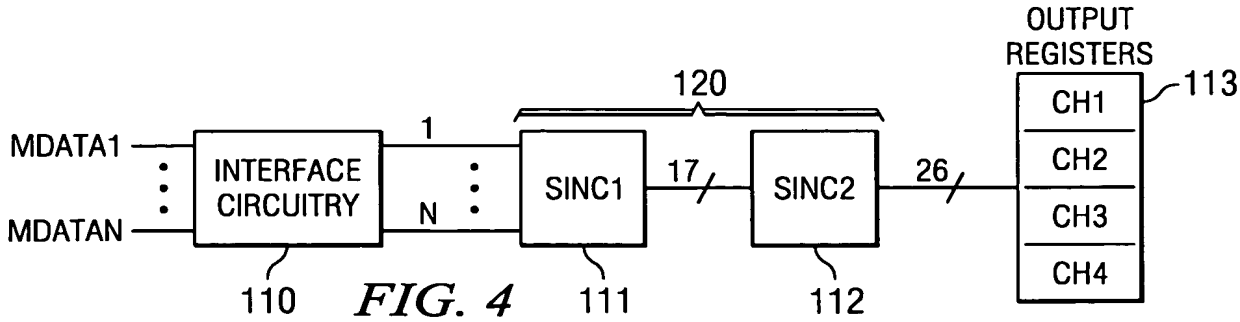


FIG. 4

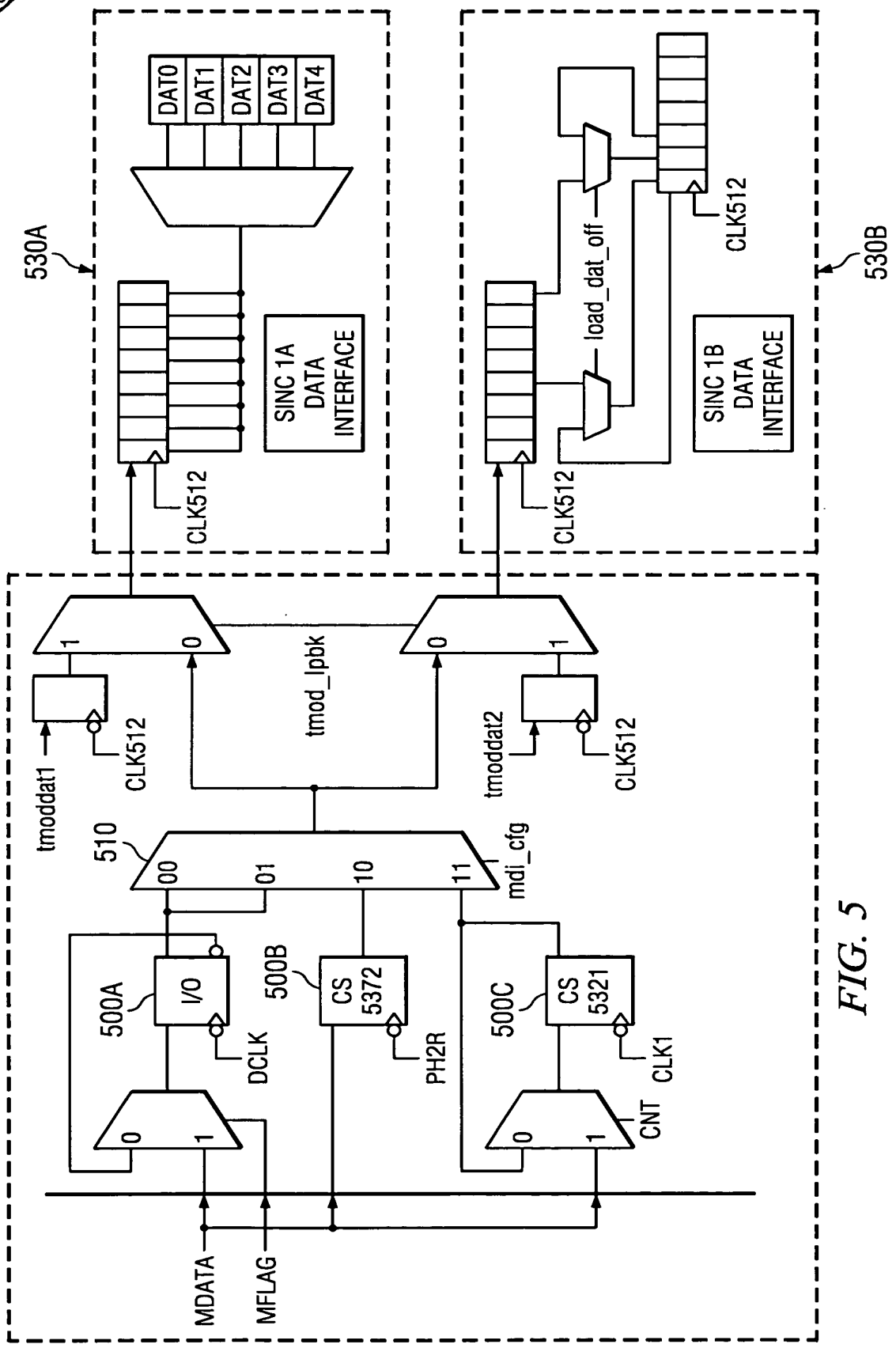
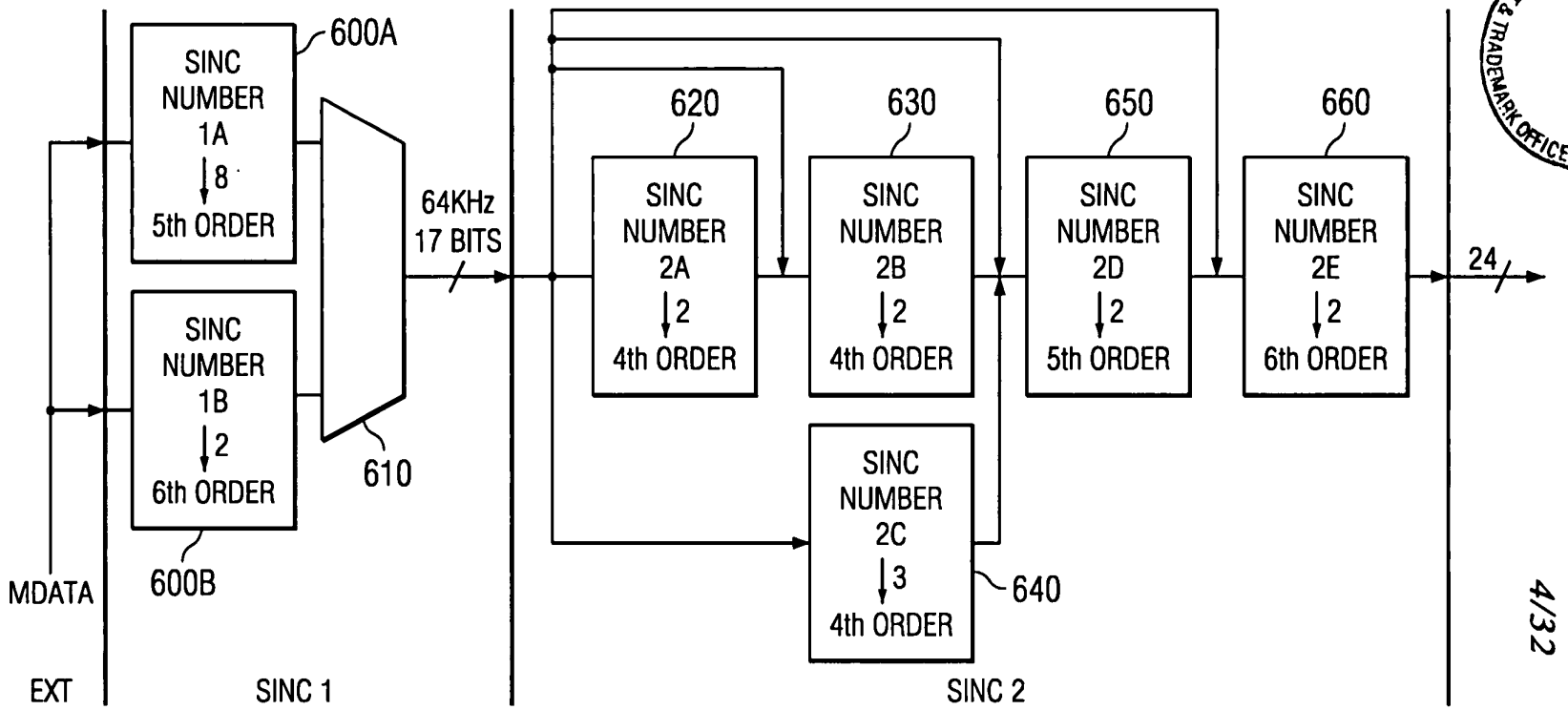


FIG. 5

FIG. 6



4/32

FIG. 7

• FIFTH ORDER DECIMATE BY 8:

$$H(z) = \left(\frac{1-z^{-8}}{1-z^{-1}} \right)^5$$

• 36 TAP FIR FILTER. HALF OF THE (SYMMETRIC) COEFFICIENTS

$h_0=1$	$h_1=5$	$h_2=15$	$h_3=35$	$h_4=70$	$h_5=126$	$h_6=210$	$h_7=330$	$h_8=490$
$h_9=690$	$h_{10}=926$	$h_{11}=1190$	$h_{12}=1470$	$h_{13}=1750$	$h_{14}=2010$	$h_{15}=2226$	$h_{16}=2380$	$h_{17}=2460$

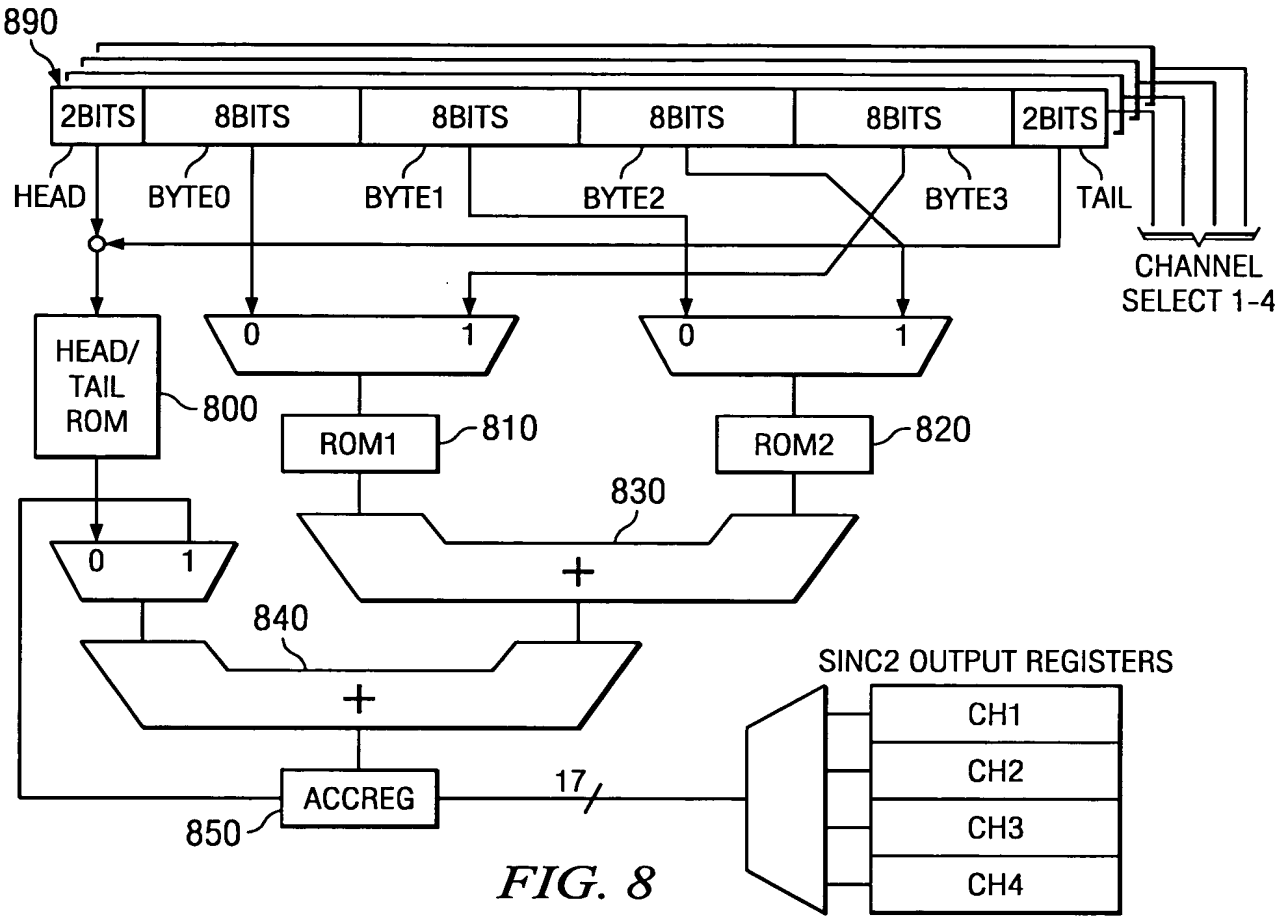


FIG. 8

$$H(z) = \left(\frac{1 - z^{-2}}{1 - z^{-1}} \right)^{6}$$

IMPULSE RESPONSE:

$$y[n] = x[n] + 6 \cdot x[n-1] + 15 \cdot x[n-2] + 20 \cdot x[n-3] + 15 \cdot x[n-4] + 6 \cdot x[n-5] + x[n-6]$$

FIG. 9

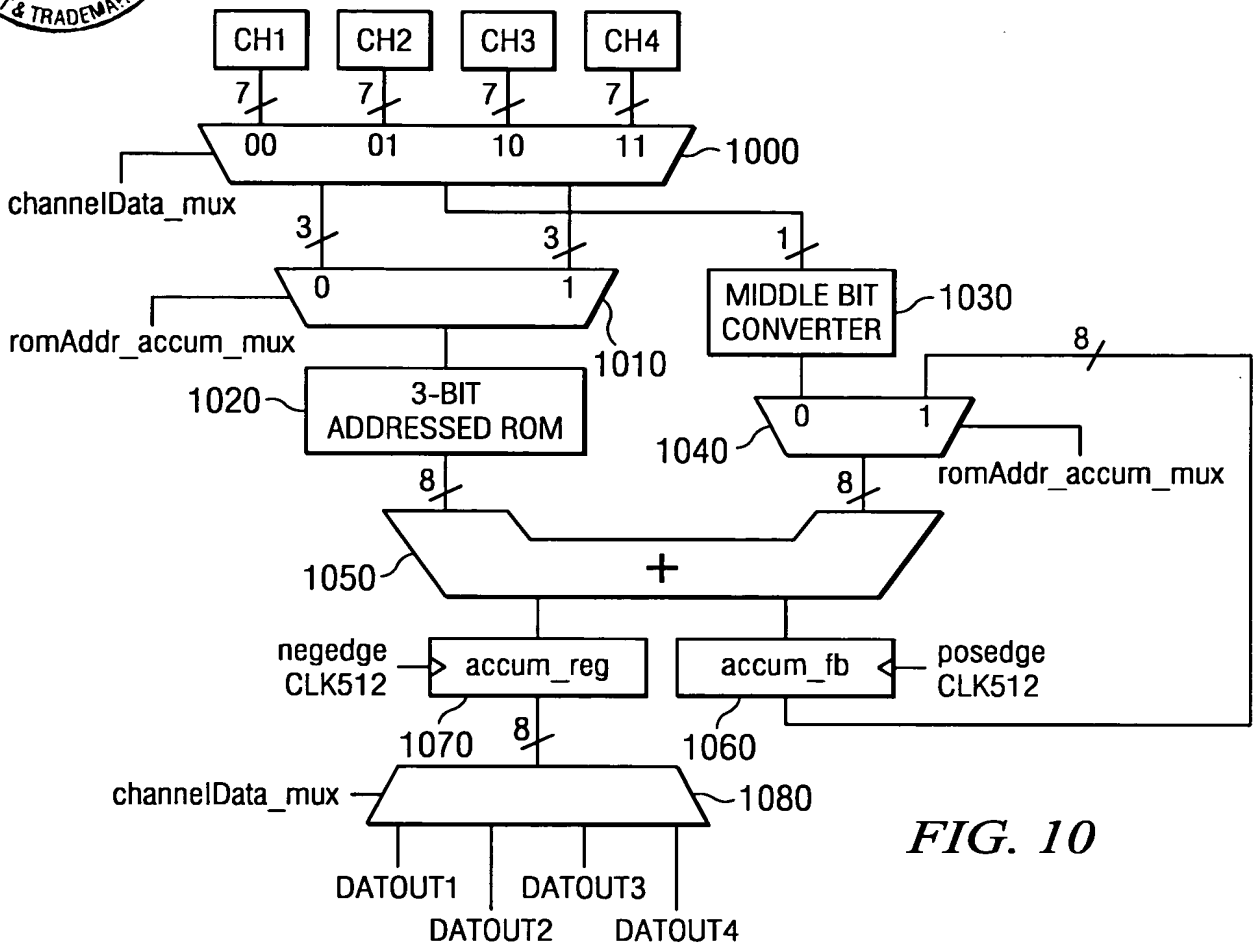


FIG. 10

FILTER NAME	SYSTEM FUNCTION	IMPULSE RESPONSE (FILTER COEFFICIENTS)
Sinc2(a) Sinc2(b)	$H(z)=\left(\frac{1-z^{-2}}{1-z^{-1}}\right)^4$	h[n]=[1 4 6 4 1]
Sinc2(c)	$H(z)=\left(\frac{1-z^{-3}}{1-z^{-1}}\right)^4$	h[n]=[1 4 10 16 19 16 10 4 1]
Sinc2(d)	$H(z)=\left(\frac{1-z^{-2}}{1-z^{-1}}\right)^5$	h[n]=[1 5 10 10 5 1]
Sinc2(e)	$H(z)=\left(\frac{1-z^{-2}}{1-z^{-1}}\right)^6$	h[n]=[1 6 15 20 15 6 1]

FIG. 11

Sinc2(a) and Sinc2(b):

$$y[n] = x[n] + 4x[n-1] + 6x[n-2] + 4x[n-3] + x[n-4]$$
$$= x[n] + 4x[n-1] + 4x[n-2] + 2x[n-2] + 4x[n-3] + x[n-4]$$

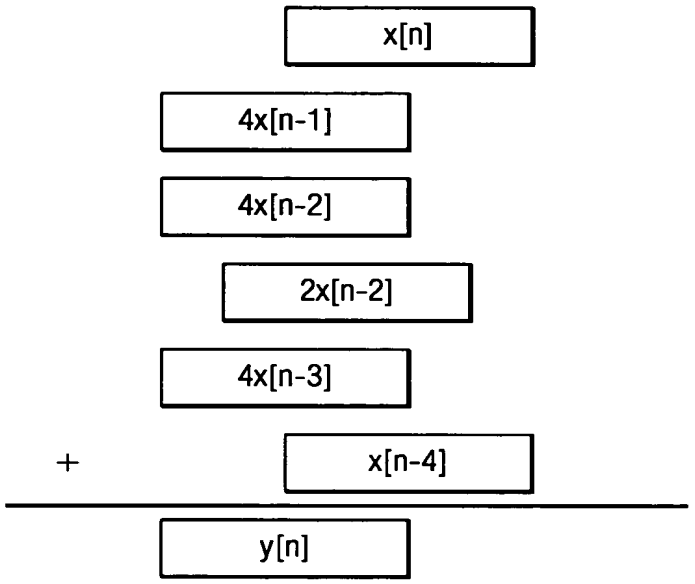


FIG. 12

Sinc2(a) and Sinc2(b):

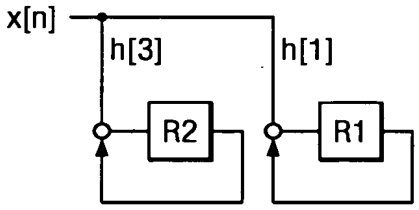


FIG. 14A

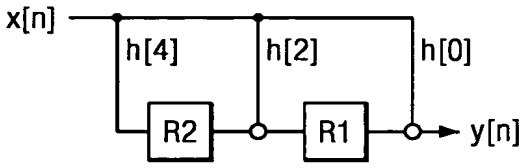


FIG. 14B

Sinc2(d):

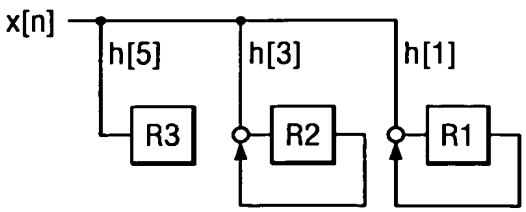


FIG. 15A

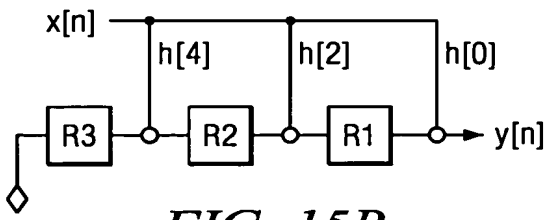


FIG. 15B

FIG. 13A

Sinc2(c):

$$y[n] = x[n] + 4x[n-1] + 10x[n-2] + 16x[n-3] + 19x[n-4] + 16x[n-5] + 10x[n-6] + 4x[n-7] + x[n-8]$$
$$= x[n] + 4x[n-1] + [8x[n-2] + 2x[n-2]] + 16x[n-3] + [16x[n-4] + 2x[n-4] + x[n-4]]$$
$$+ 16x[n-5] + [8x[n-6] + 2x[n-6]] + 4x[n-7] + x[n-8]$$

FIG. 13B

Sinc2(d):

$$y[n] = x[n] + 5x[n-1] + 10x[n-2] + 10x[n-3] + 5x[n-4] + x[n-5]$$
$$= x[n] + [4x[n-1] + x[n-1]] + [8x[n-2] + 2x[n-2]] + [8x[n-3] + 2x[n-3]] + [4x[n-4] + x[n-4]] + x[n-5]$$

FIG. 13C

Sinc2(e):

$$y[n] = x[n] + 6x[n-1] + 15x[n-2] + 20x[n-3] + 15x[n-4] + 6x[n-5] + x[n-6]$$
$$= x[n] + [4x[n-1] + 2x[n-1]] + [16x[n-2] - x[n-2]] + [16x[n-3] + 4x[n-3]]$$
$$+ [16x[n-4] - x[n-4]] + [4x[n-5] + 2x[n-5]] + x[n-6]$$

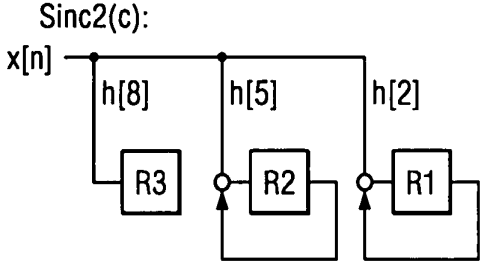


FIG. 16A

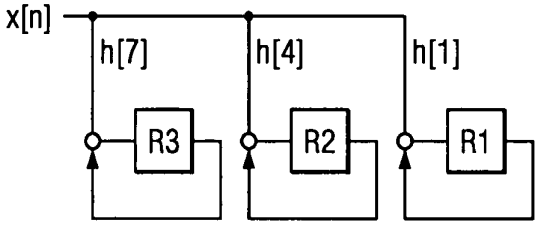


FIG. 16B

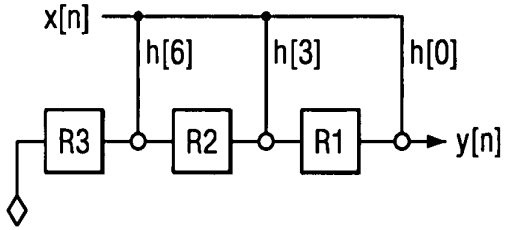


FIG. 16C

Sinc2(e):

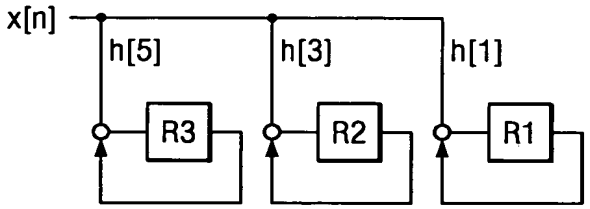


FIG. 17A

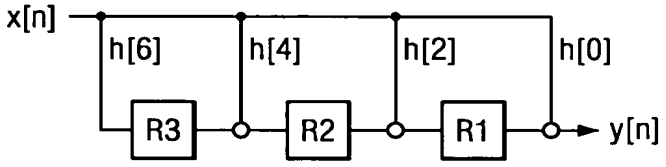


FIG. 17B

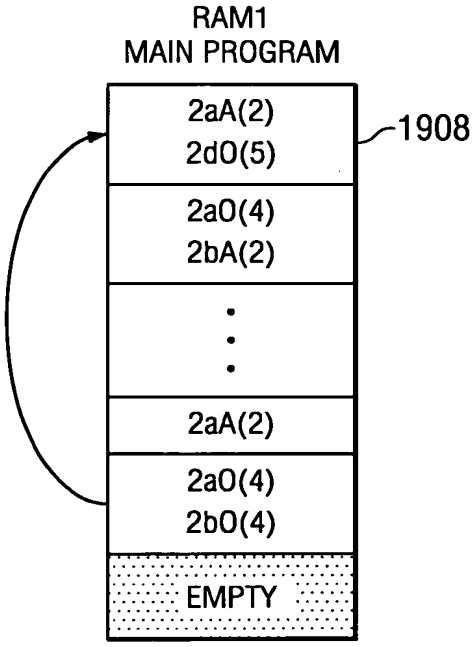


FIG. 19A

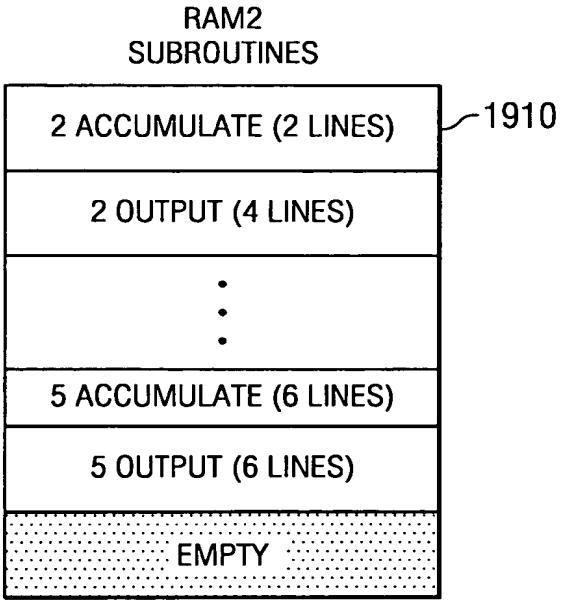


FIG. 19B

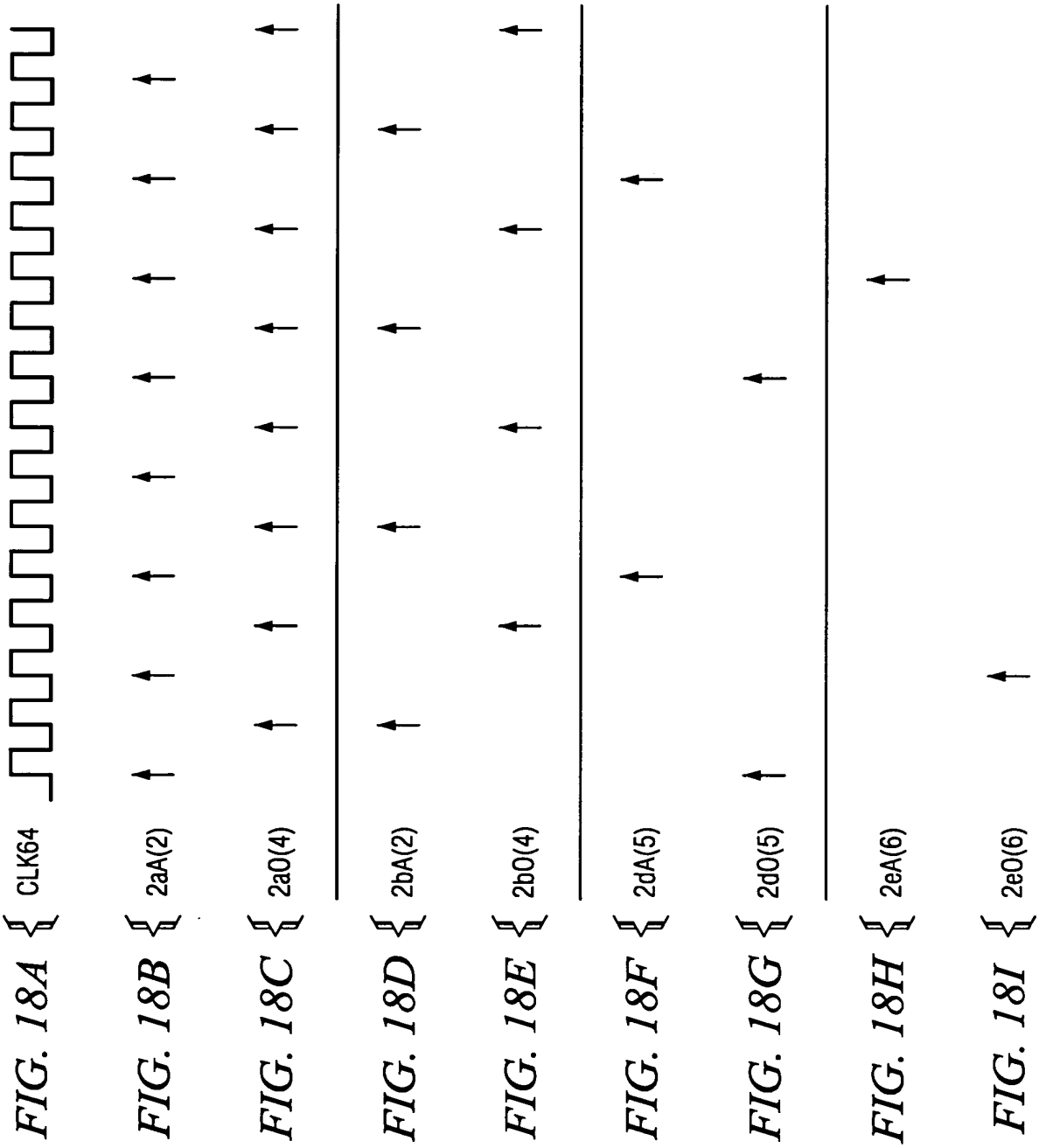


FIG. 18A

FIG. 18B

FIG. 18C

FIG. 18D

FIG. 18E

FIG. 18F

FIG. 18G

FIG. 18H

FIG. 18I

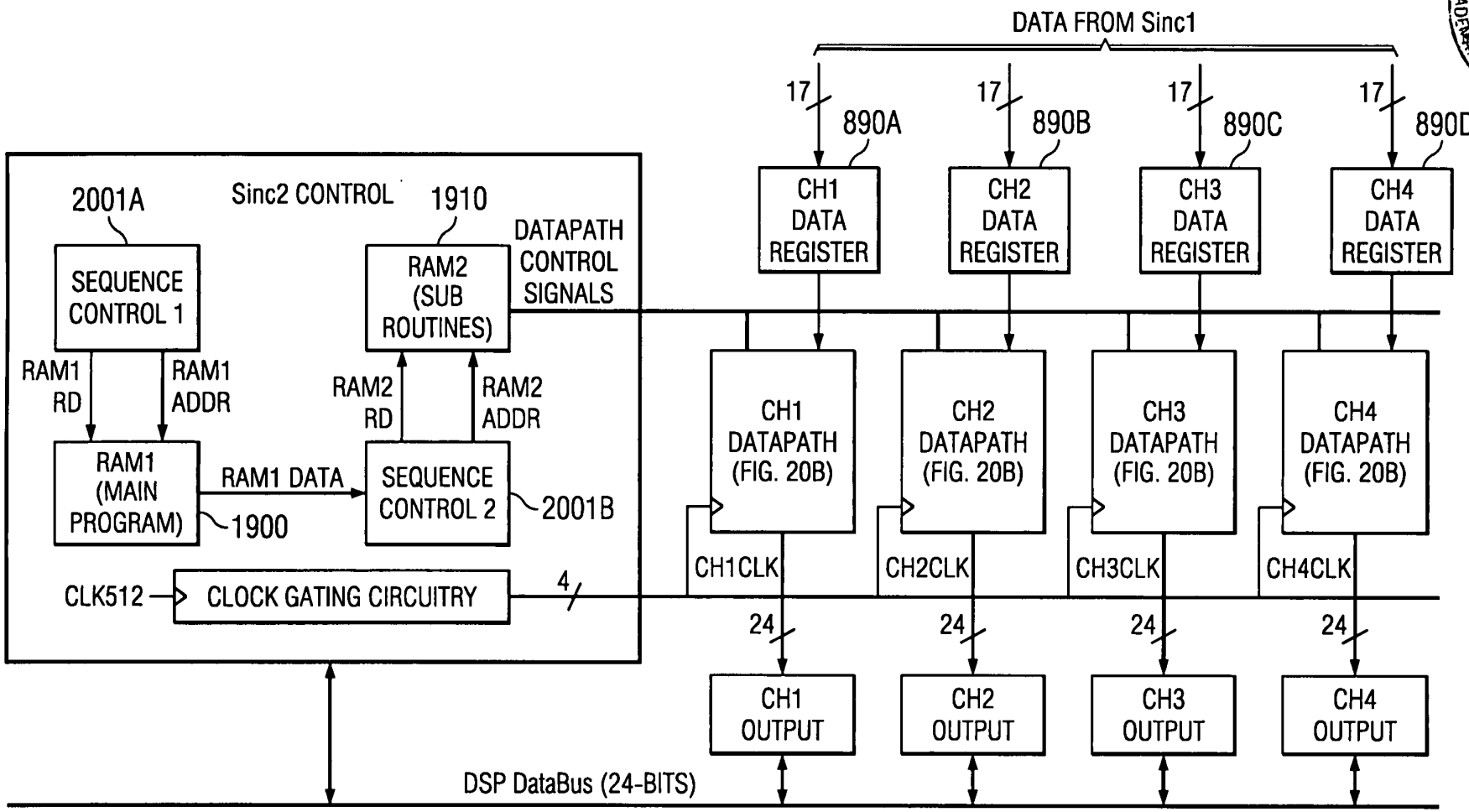


FIG. 20A

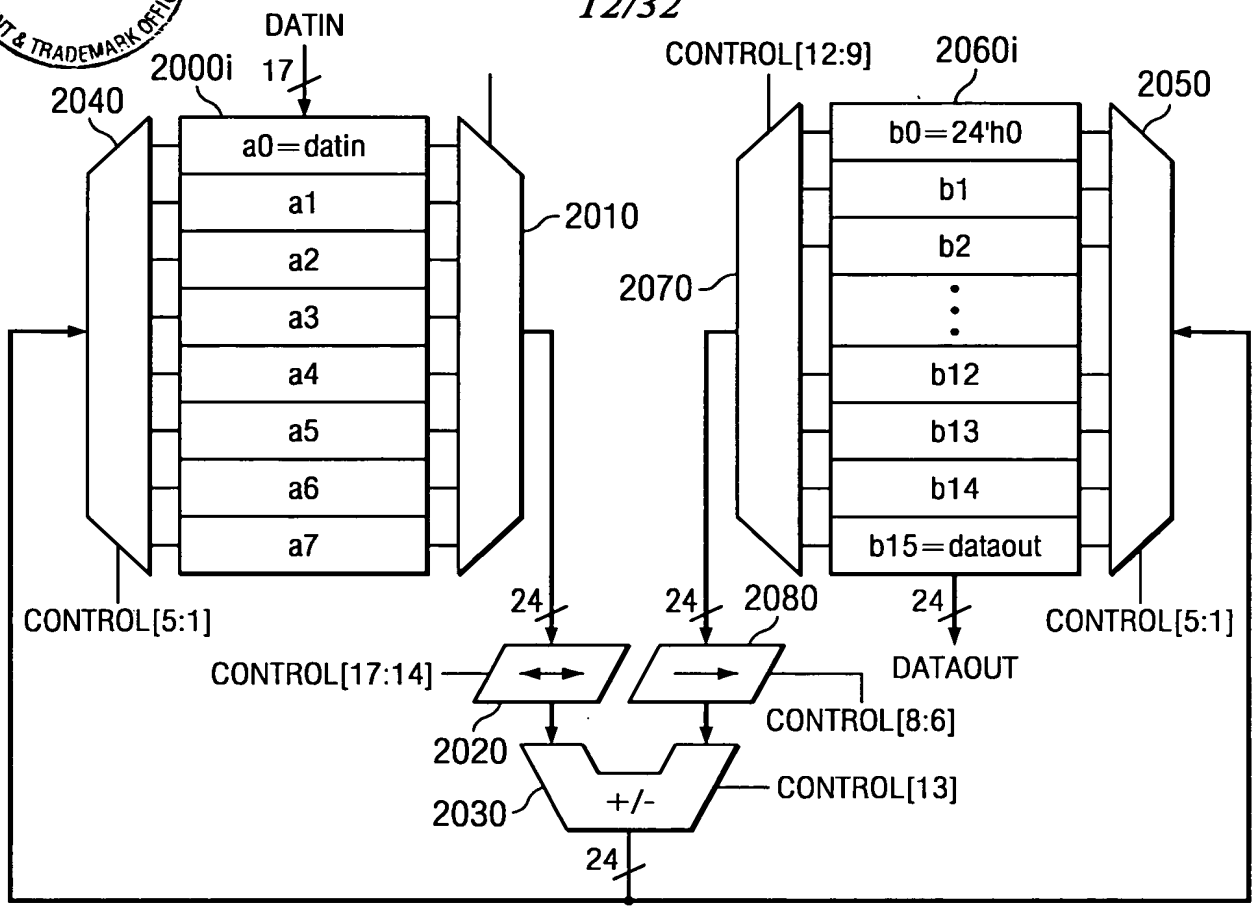


FIG. 20B

PROGRAMMING PROCEDURE:

1. SELECT DECIMATION RATE.

2. SELECT REQUIRED MINI-SINCS AND ASSOCIATAED ACCUMULATE AND OUTPUT SUBROUTINES.

3. SEPARATE COEFFICIENTS INTO FORM SUITABLE FOR SHIFT-ADD OPERATIONS.

4. CHECK FOR OVERFLOW AFTER EACH ADDITION IN THE FILTER.

5. PERFORM NECESSARY TRUNCATION TO 24 BITS AND SCALING OF SUBSEQUENT COEFFICIENTS IN MINI-SINCS.

6. TIME MULTIPLEX ACCUMULATE AND OUTPUT SUBROUTINES SO THAT A MAXIMUM OF 8 ADDITIONS/SUBTRACTIONS ARE PERFORMED FOR EACH INPUT FROM SINC1.

7. CREATE CODE FOR RAM2 (ACCUMULATE AND OUTPUT SUBROUTINES) IN THE FORM: [Coeff 1] [Src 1] [Src 2] [Dest] [Coeff2] [Done Subroutine]

8. CREATE CODE FOR RAM1 (MAIN CONTROL CODE)
[Line #] [Wait for new data] [Done program]

FIG. 21

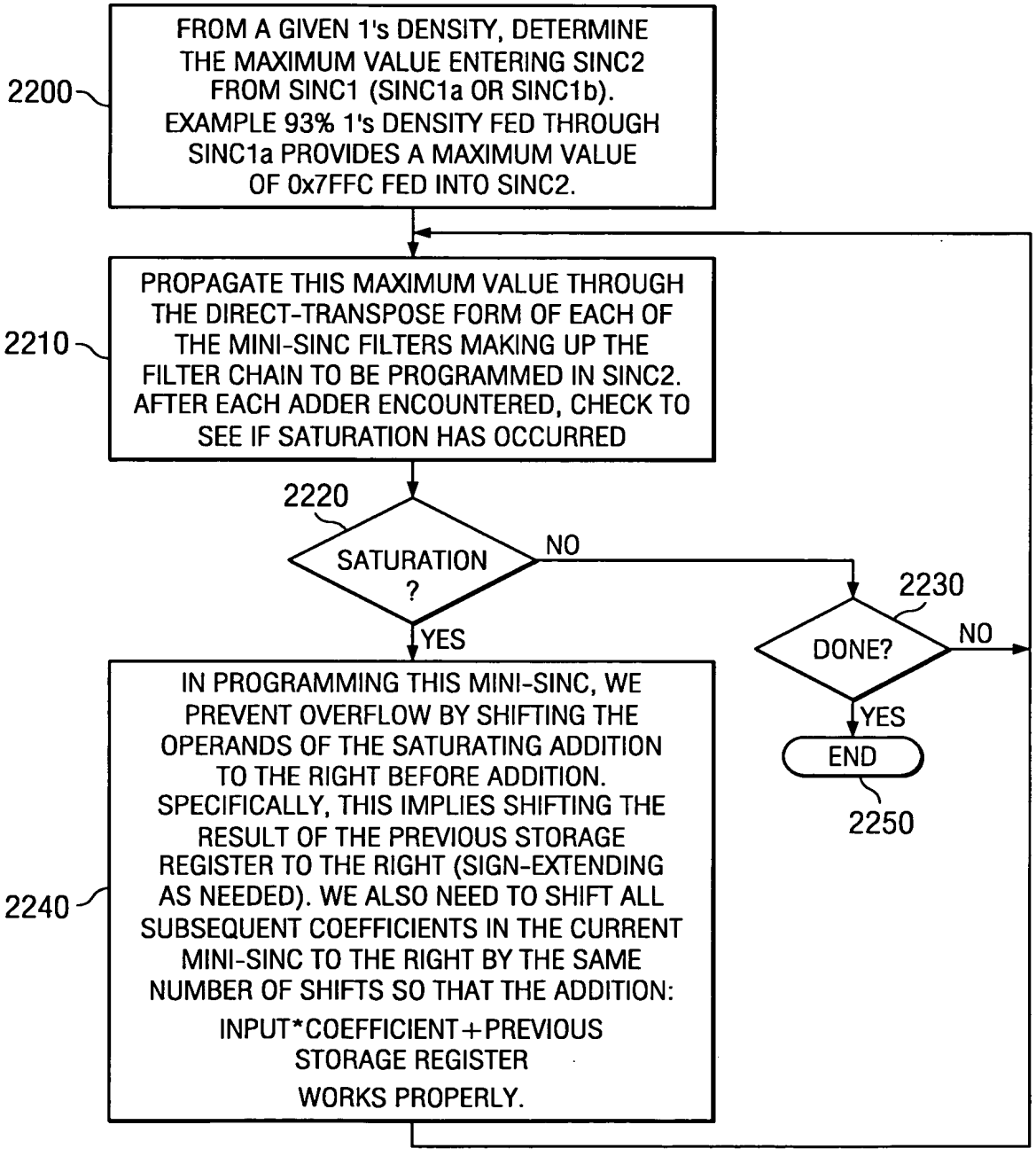


FIG. 22

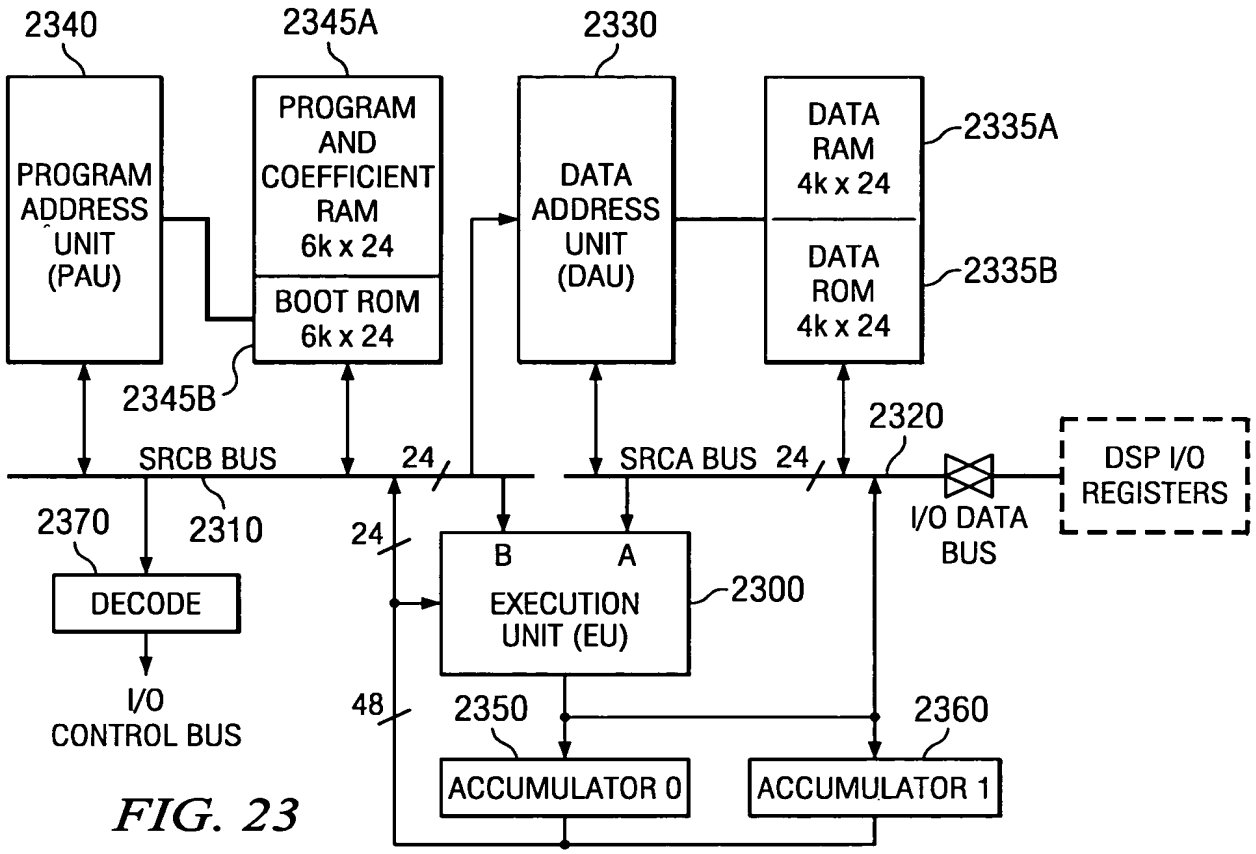


FIG. 23

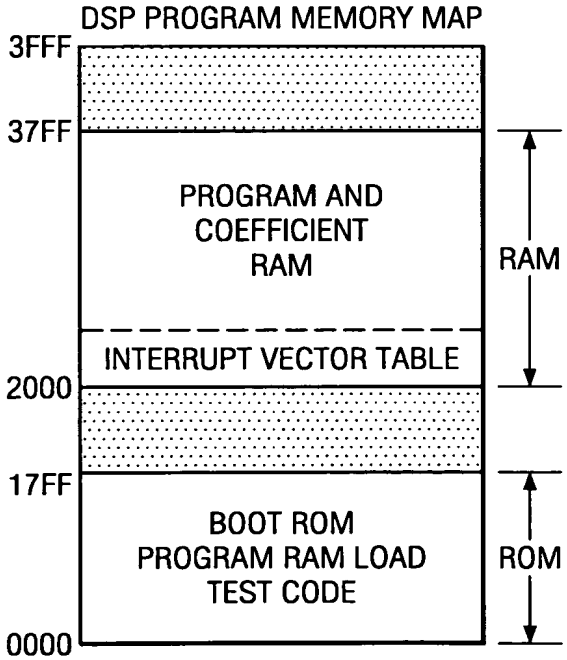


FIG. 24A

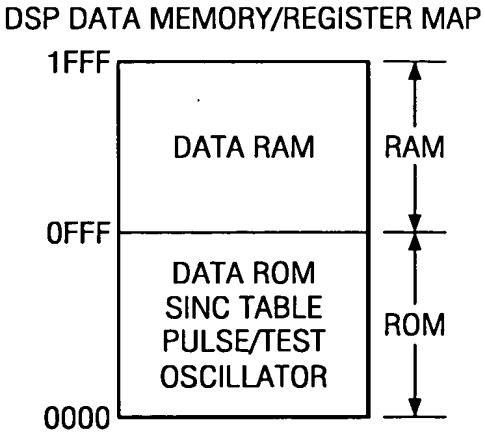


FIG. 24B

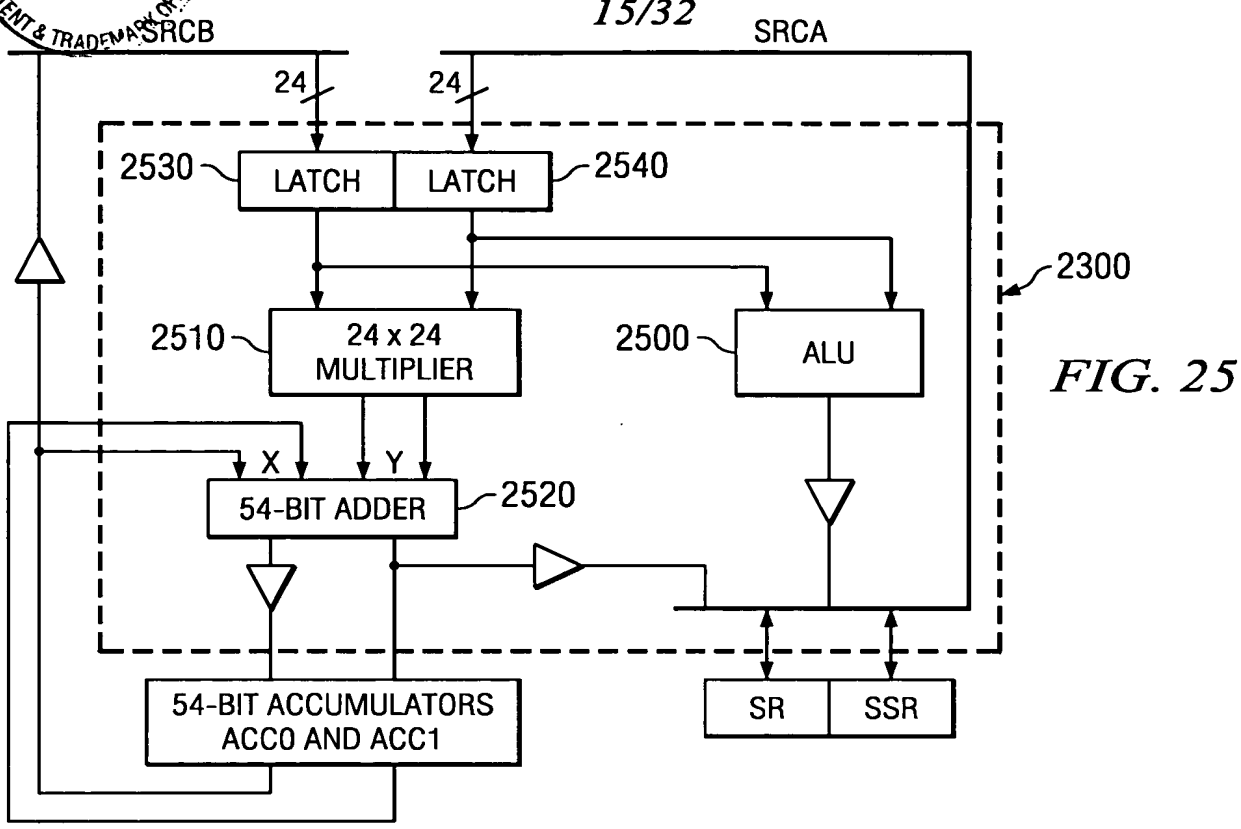


FIG. 25

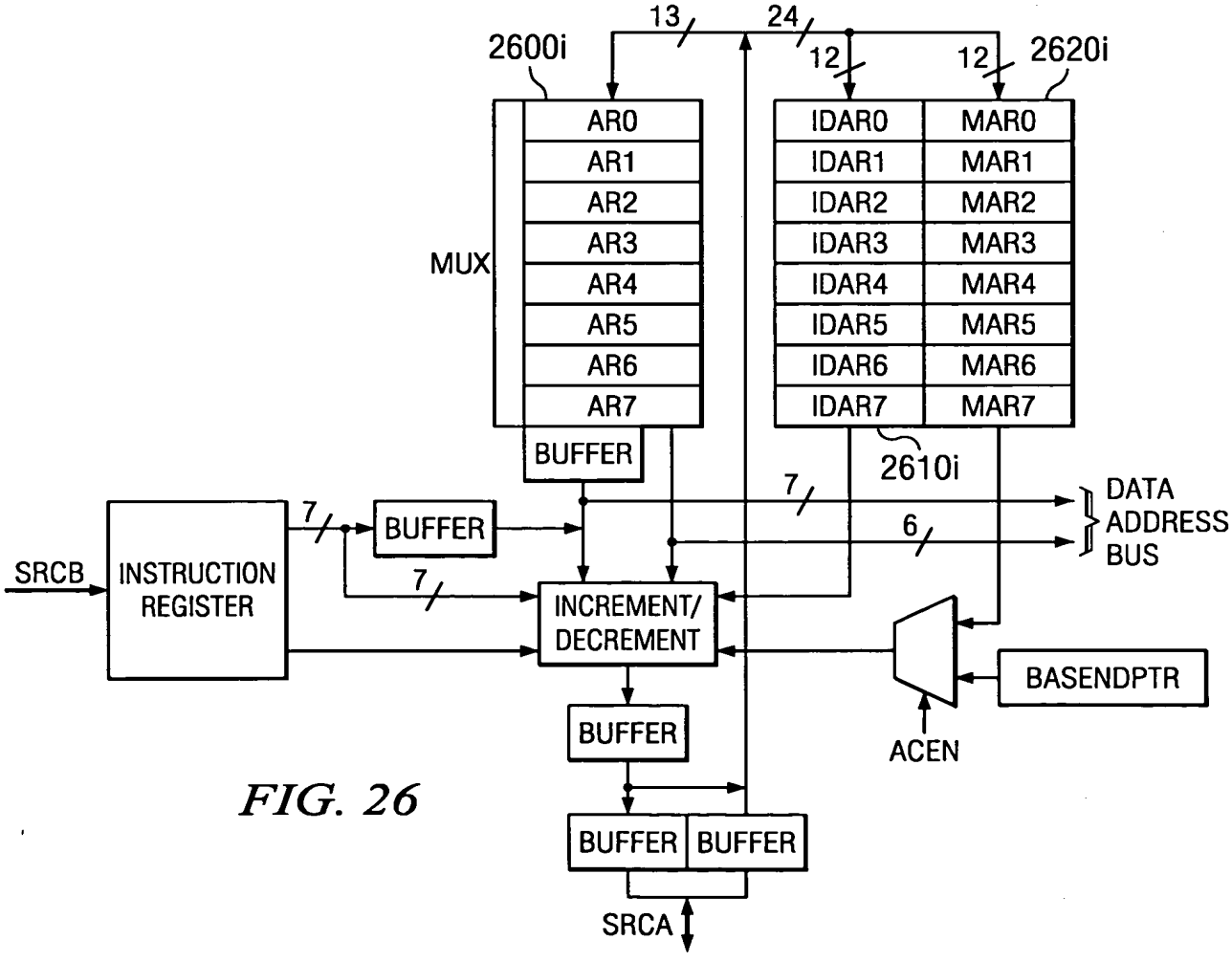


FIG. 26

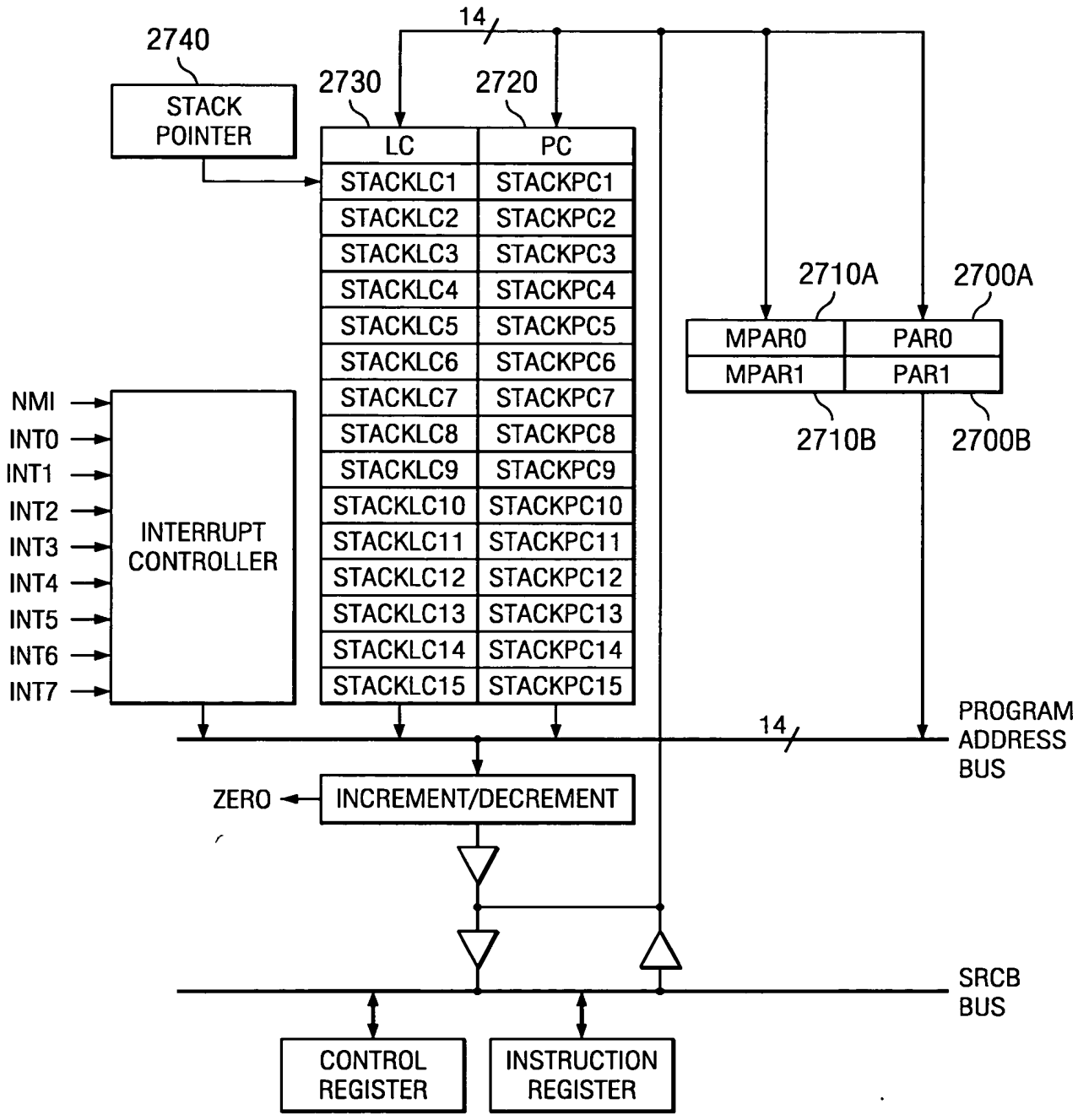


FIG. 27

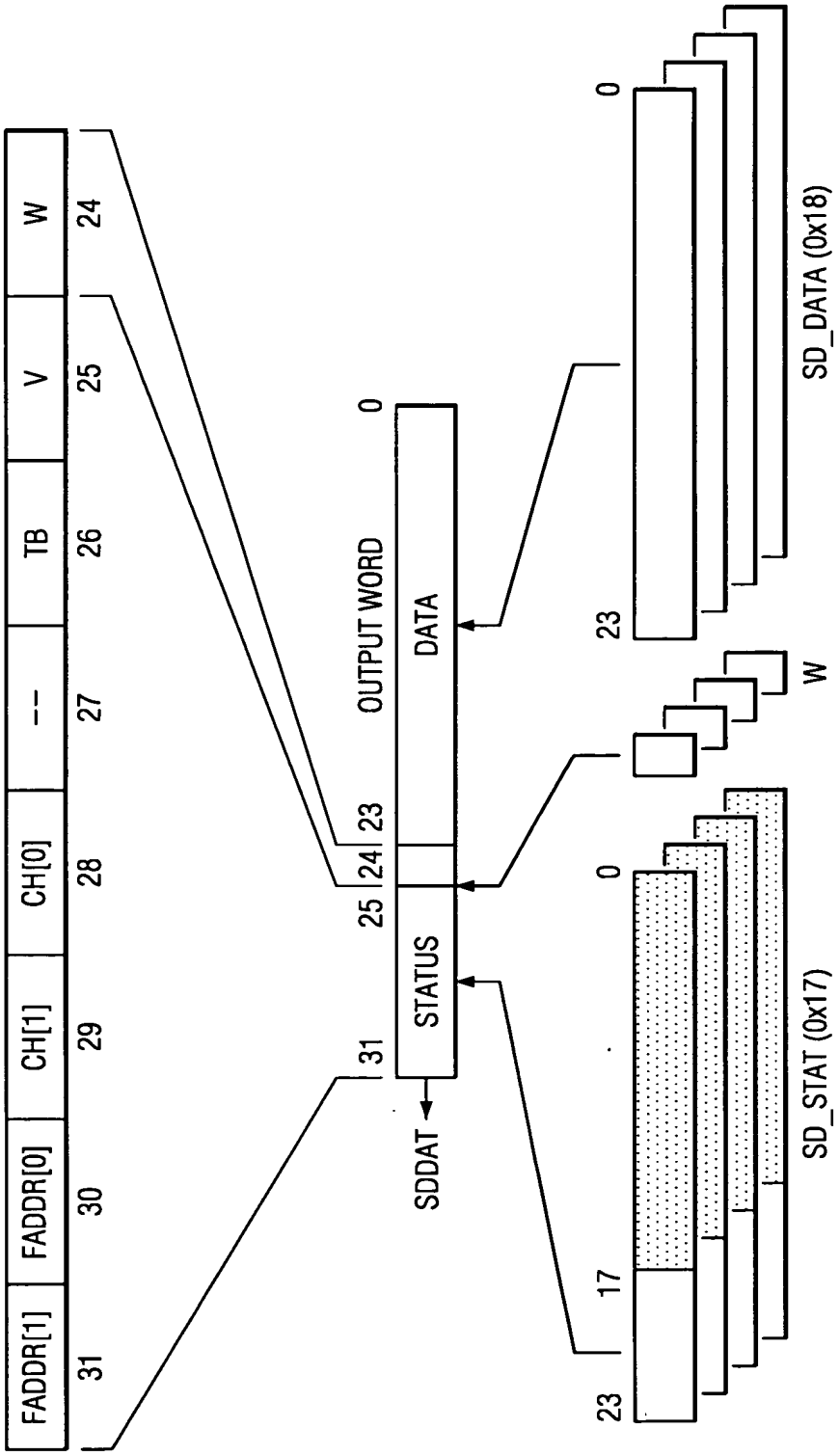


FIG. 28

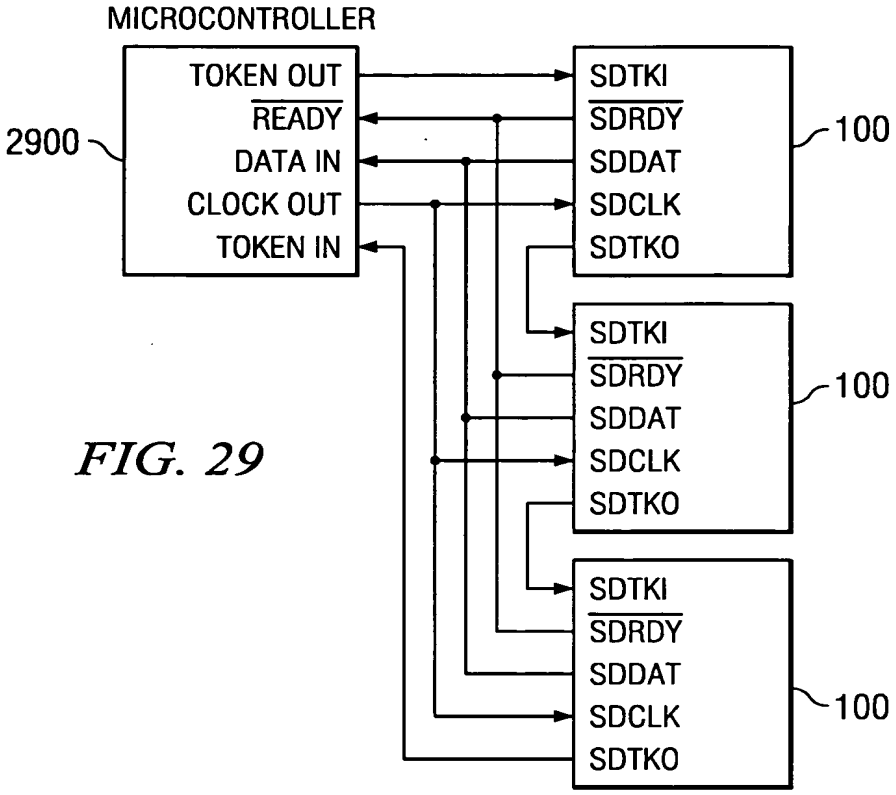


FIG. 29

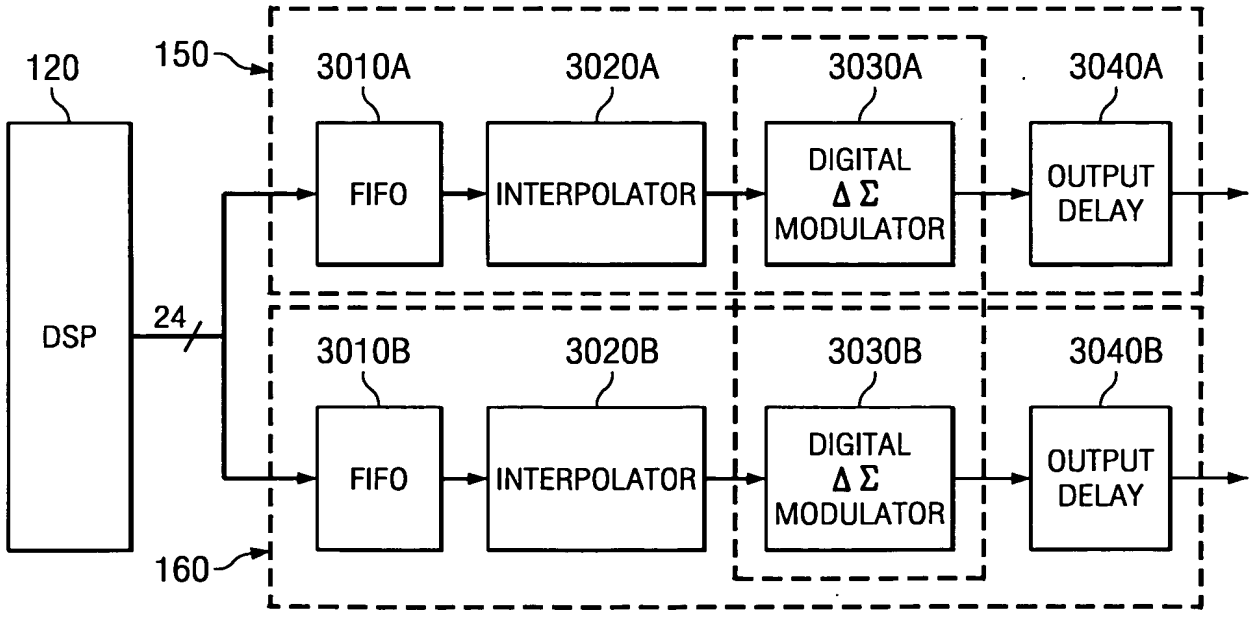


FIG. 30A

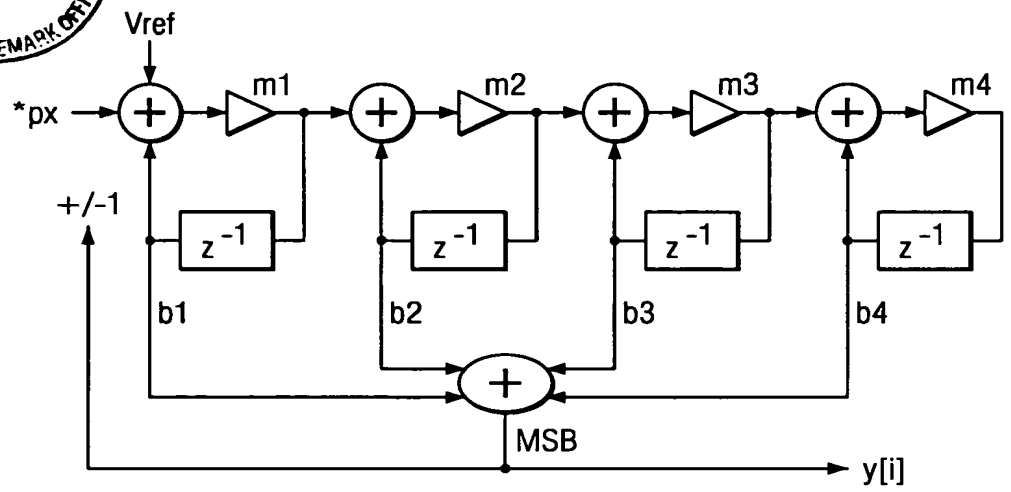


FIG. 30B

FIG. 30C-1 — WIRE

FIG. 30C-2 — 24 WIRES

FIG. 30C-3 — REGISTER

FIG. 30C-4 — MULTIPLEXER

FIG. 30C-5 — TRISTATE BUFFER

FIG. 30C-6 — INVERTER

FIG. 30C-7 — EXCLUSIVE OR GATE

FIG. 30C-8 — ADDER

FIG. 30C-9 — MULTIPLIER

FIG. 30C-10 — RIGHT SHIFTER

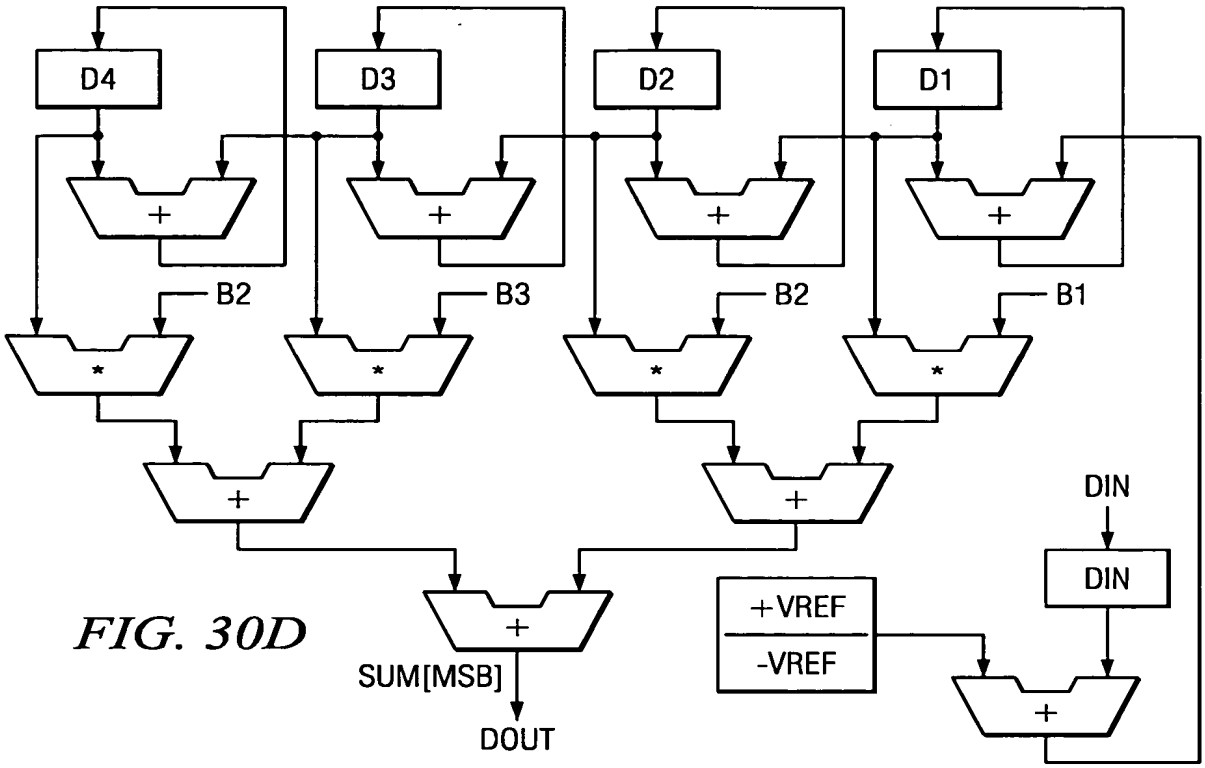


FIG. 30D

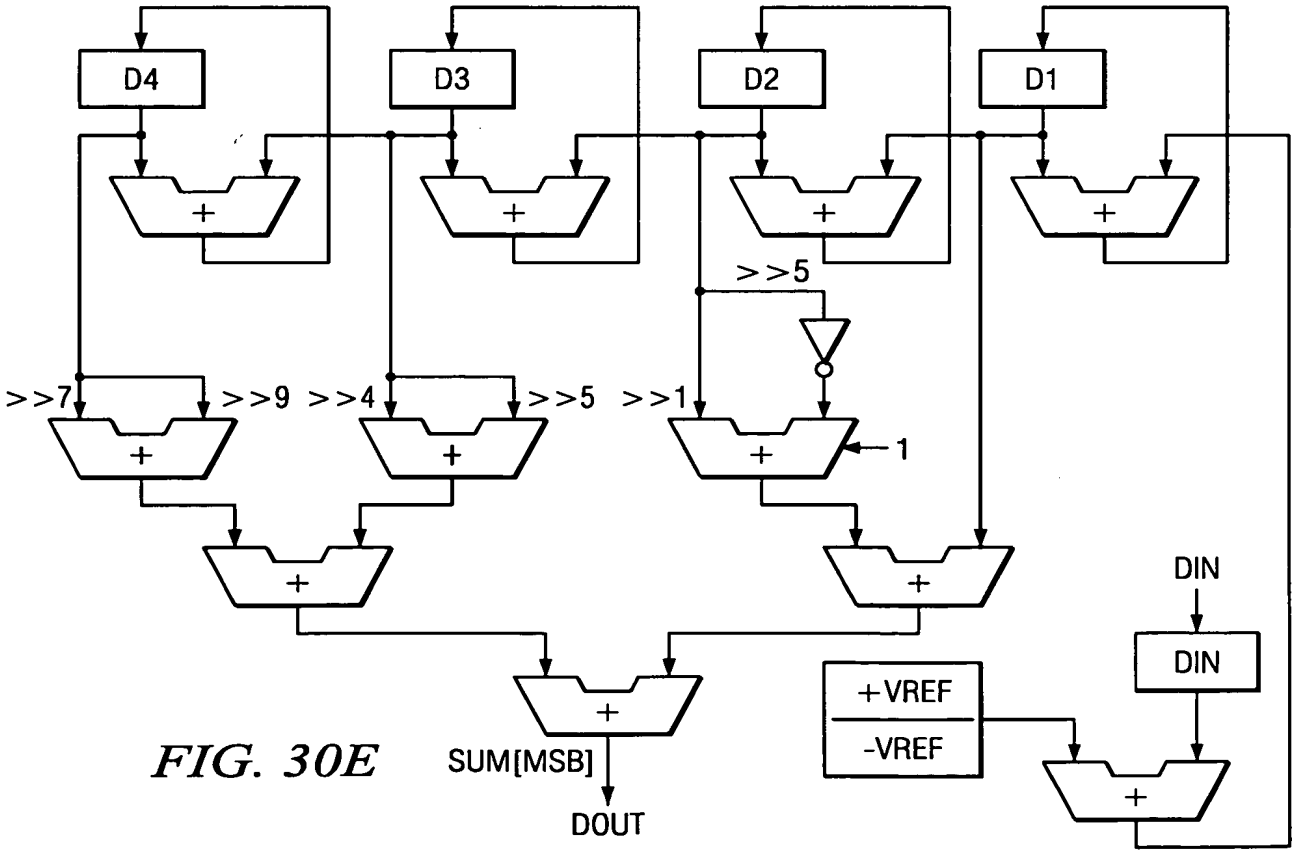


FIG. 30E

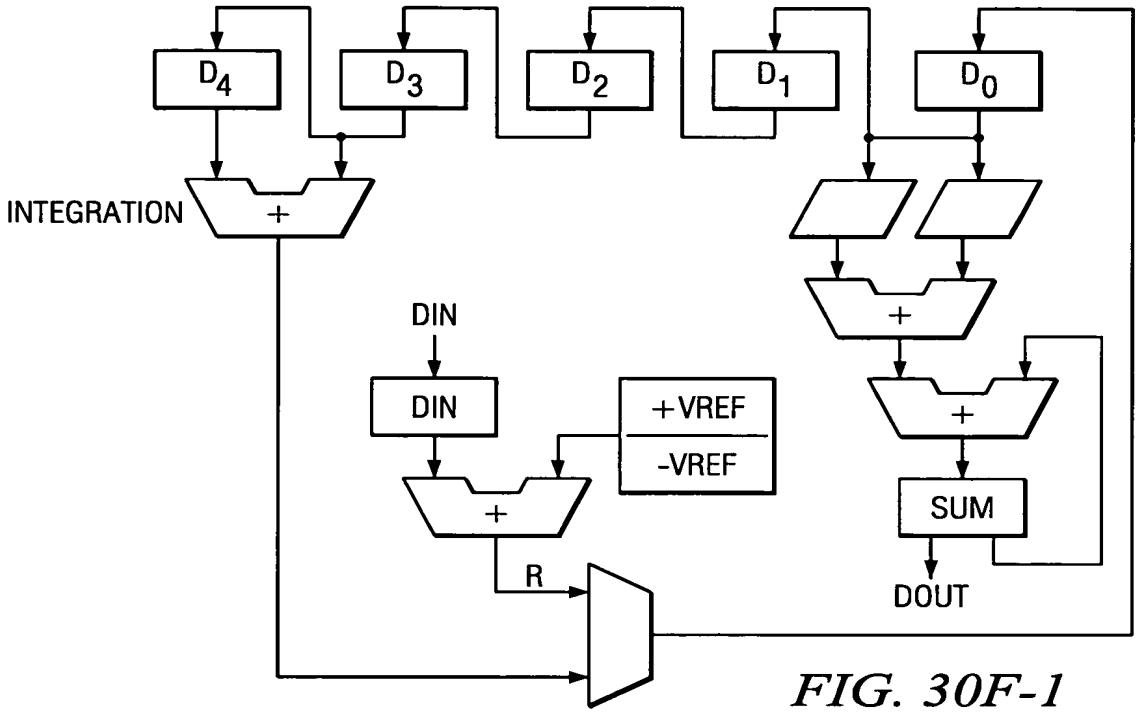


FIG. 30F-1

STATE	ACTIONS DURING STATE		
S0	$D_0(D4_k) = D_4(D4_{k-1}) + D_3(D3_{k-1})$	CLEAR SUM	LOAD DIN_k
S1	$D_0(D3_k) = D_4(D3_{k-1}) + D_3(D2_{k-1})$	$SUM_k += D_0(D4_k) >> Shift4$	
S2	$D_0(D2_k) = D_4(D2_{k-1}) + D_3(D1_{k-1})$	$SUM_k += D_0(D3_k) >> Shift3$	
S3	$D_0(D1_k) = D_4(D1_{k-1}) + D_3(R_{k-1})$	$SUM_k += D_0(D2_k) >> Shift2$	
S4		$SUM_k += D_0(D1_k) >> Shift1$	
S5	$D_0(R_k) = DIN_k +/- VREF$		

FIG. 30F-2

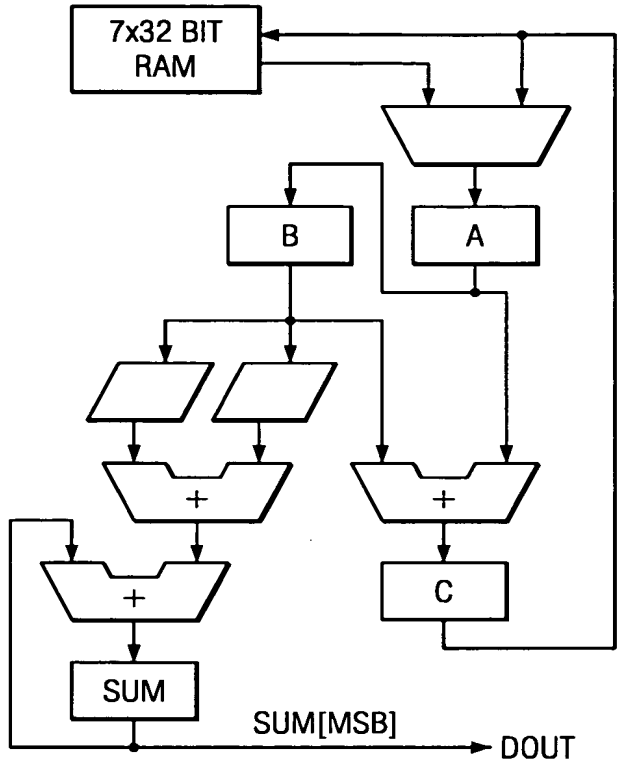


FIG. 30G-1

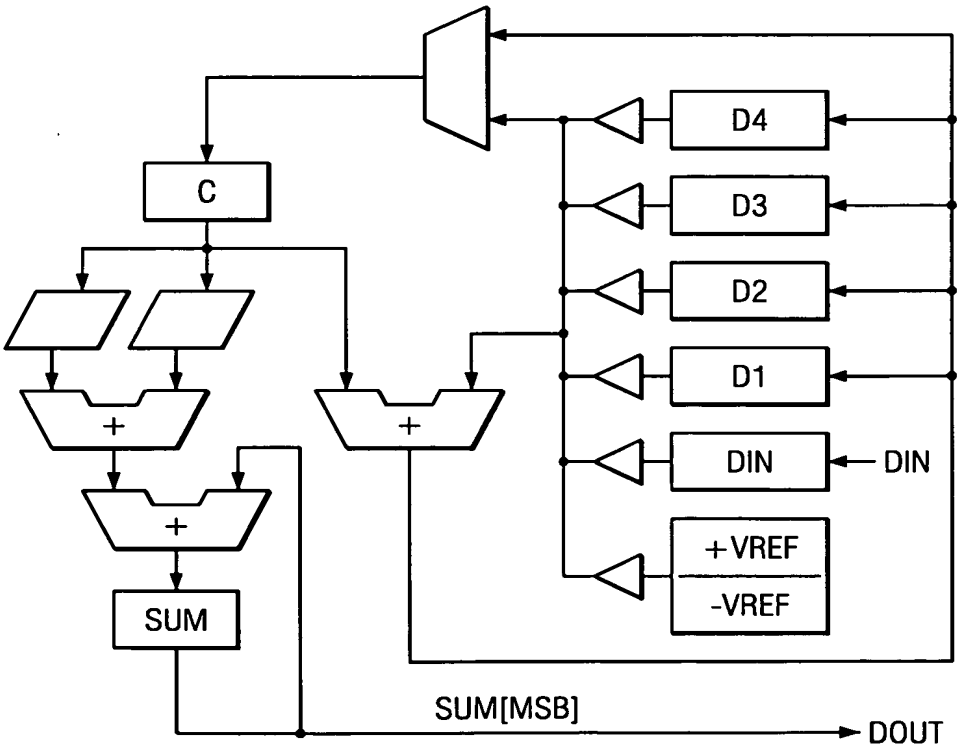


FIG. 30H-1

STATE	ACTIONS DURING STATE			
S0	CLEAR SUM	CLEAR C	CLEAR B	CLEAR A
S1				LOAD A<Mem($D4_k$)
S2			SHIFT B<A($D4_k$)	LOAD A<Mem($D3_k$)
S3	SUM_k+= B($D4_k$)>>Shift4	C = B($D4_k$) + A($D3_k$)	SHIFT B<A($D3_k$)	LOAD A<Mem($D2_k$)
S4				STORE C>Mem($D4_{k+1}$)
S5	SUM_k+= B($D3_k$)>>Shift3	C = B($D3_k$) + A($D2_k$)	SHIFT B<A($D2_k$)	LOAD A<Mem($D1_k$)
S6				STORE C>Mem($D3_{k+1}$)
S7	SUM_k+= B($D2_k$)>>Shift2	C = B($D2_k$) + A($D1_k$)	SHIFT B<A($D1_k$)	LOAD A<Mem(DIN_k)
S8				STORE C>Mem($D2_{k+1}$)
S9	SUM_k+= B($D1_k$)>>Shift1	C = B($D1_k$) + A(DIN_k)	SHIFT B<A(DIN_k)	LOAD A<Mem(VREF)
S10			SHIFT B<A(VREF)	LOADREG A<C(TEMP)
S11		C = +/- B(VREF) + A(TEMP)		
S12				STORE C>Mem($D1_{k+1}$)

FIG. 30G-2

STATE	ACTIONS DURING STATE			
S0	CLEAR SUM	LOAD C < $D4_k$		LOAD DIN_k
S1	SUM_k+= C($D4_k$)>>Shift4	LOAD C < $D3_k$	$D4_{k+1}$ = C($D4_k$) + $D3_k$	
S2	SUM_k+= C($D3_k$)>>Shift3	LOAD C < $D2_k$	$D3_{k+1}$ = C($D3_k$) + $D2_k$	
S3	SUM_k+= C($D2_k$)>>Shift2	LOAD C < $D1_k$	$D2_{k+1}$ = C($D2_k$) + $D1_k$	
S4	SUM_k+= C($D1_k$)>>Shift1	C(TEMP) = C($D1_k$) + DIN_k		
S5			$D1_{k+1}$ = C(TEMP) +/- VREF	

FIG. 30H-2

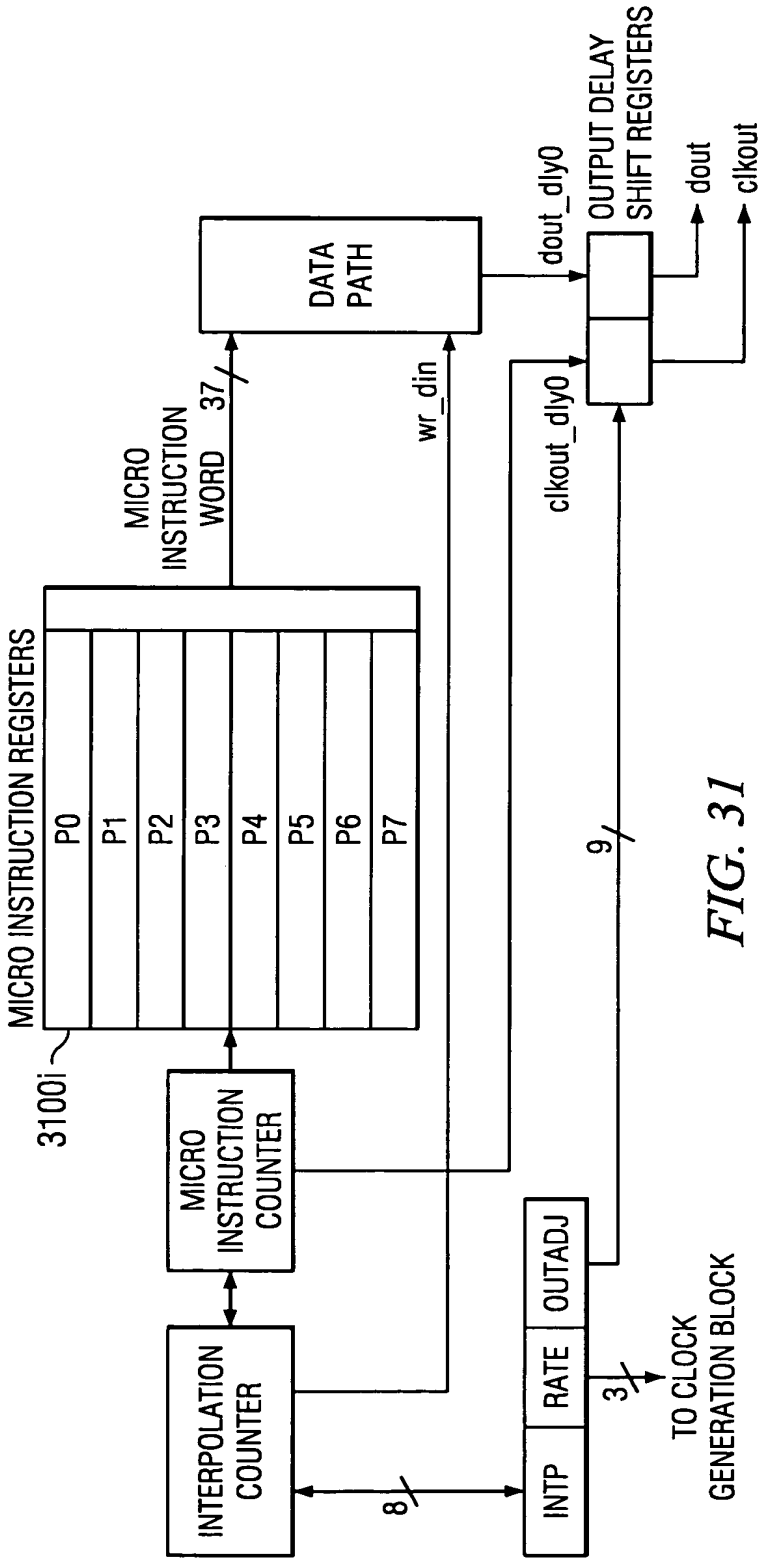


FIG. 31

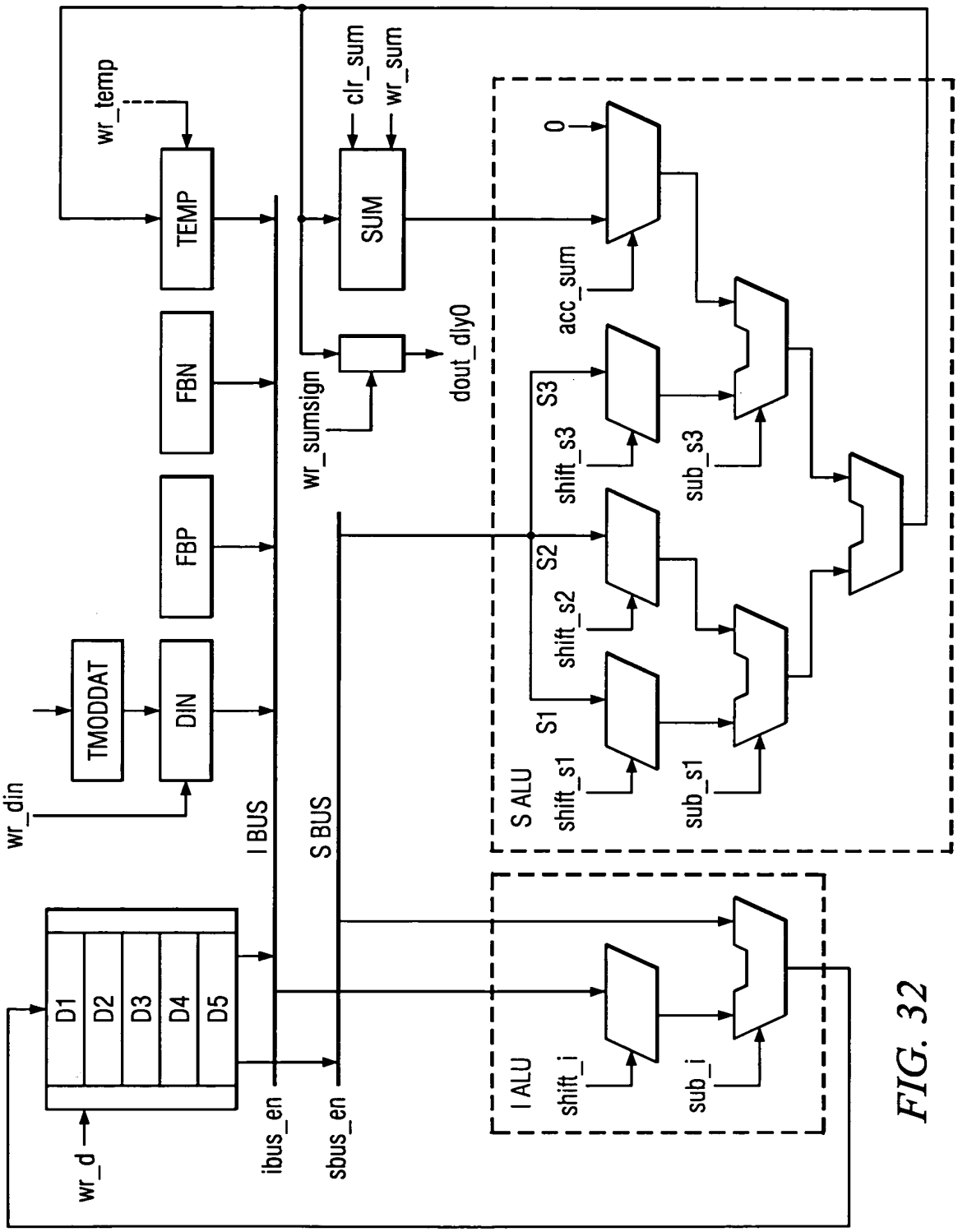


FIG. 32

P	Feedforward	INTEGRATION	TEMP	DIN	SUM	SUMSIGN	TEMP	S BUS	I BUS	WRITE I
0	$SUM_k = D4_k >> 11 + D4_k >> 9 + D4_k >> 7$	$D4_{k+1} = D4_k + D3_k$		LOAD DIN_k	WRITE			+D4>>7 +D4>>9 +D4>>11	+D3	D4
1	$SUM_k = SUM_k + D3_k >> 8 + D3_k >> 5 + D3_k >> 4$	$D3_{k+1} = D3_k + D2_k$			ACC./ WRITE			+D3>>4 +D3>>5 +D3>>8	+D2	D3
2	$SUM_k = SUM_k + D2_k >> 1 = D2_k >> 7 = D2_k >> 4$	$D2_{k+1} = D2_k + D1_k$			ACC./ WRITE			-D2>>4 +D2>>1 -D2>>7	+D1	D2
3	$SUM_k = SUM_k + D1_k$	$D1_{k+1'} = D1_k + DIN_k$			ACC./ WRITE	WRITE		+D1 +D1 -D1	+DIN	D1
4		$D1_{k+1} = D1_{k+1'} +/- VREF$							+FB	D1
5										
6										
7										

FIG. 33

G	C	S	T	wr_d			ibus_en			sbus_en			shift_i					s i	shift_s1					s s 1	shift_s2					shift_s3					s s 3	A	HEX
36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0	1	0	1	0	1	1	0	0	051C 00E4AC
0	0	1	0	0	1	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	1	0	0	0	0	1	04D3 0082A1
0	0	1	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	1	1	1	1	1	048A 00909F
1	0	1	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1469 000003
0	0	0	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0079 000000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 000000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 000000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000 000000

FIG. 34A

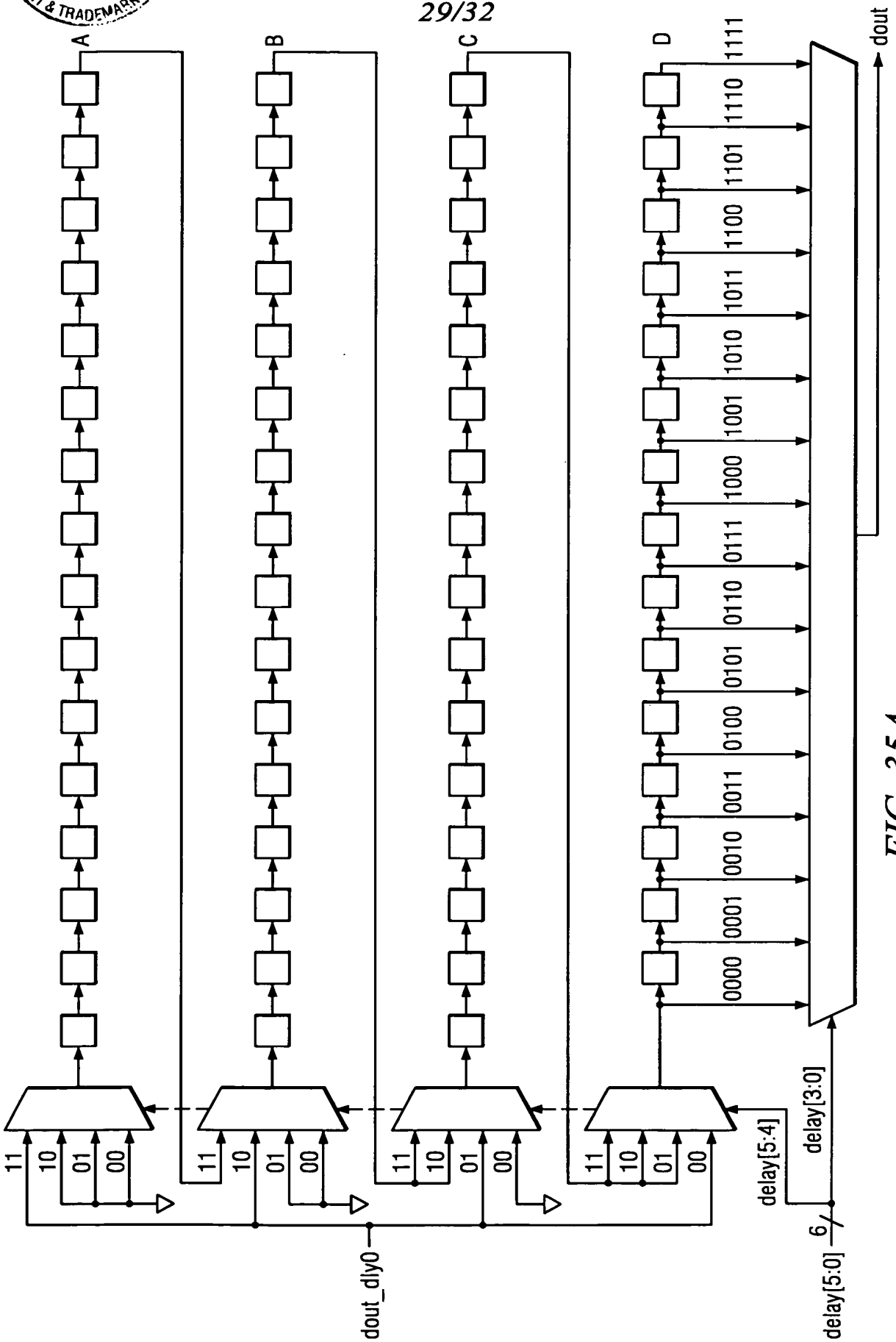


FIG. 35A

dout_dly0	DATA OUTPUT BIT, 0 DELAY
dout	DATA OUTPUT BIT, 0-63 CLOCK DELAY
delay[5:0]	HOW MANY CLOCKS (0-63) TO DELAY OUTPUT DATA dout_dly0
delay[5:4]	SELECTS SEGMENT INTO WHICH TO DIRECT dout_dly0
delay[3:0]	SELECTS WHERE TO TAP SEGMENT D TO GET dout

FIG. 35B

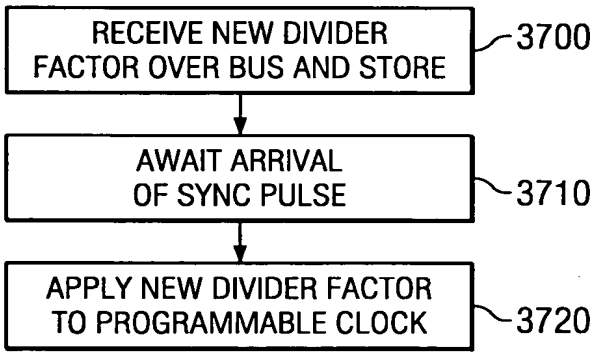


FIG. 37

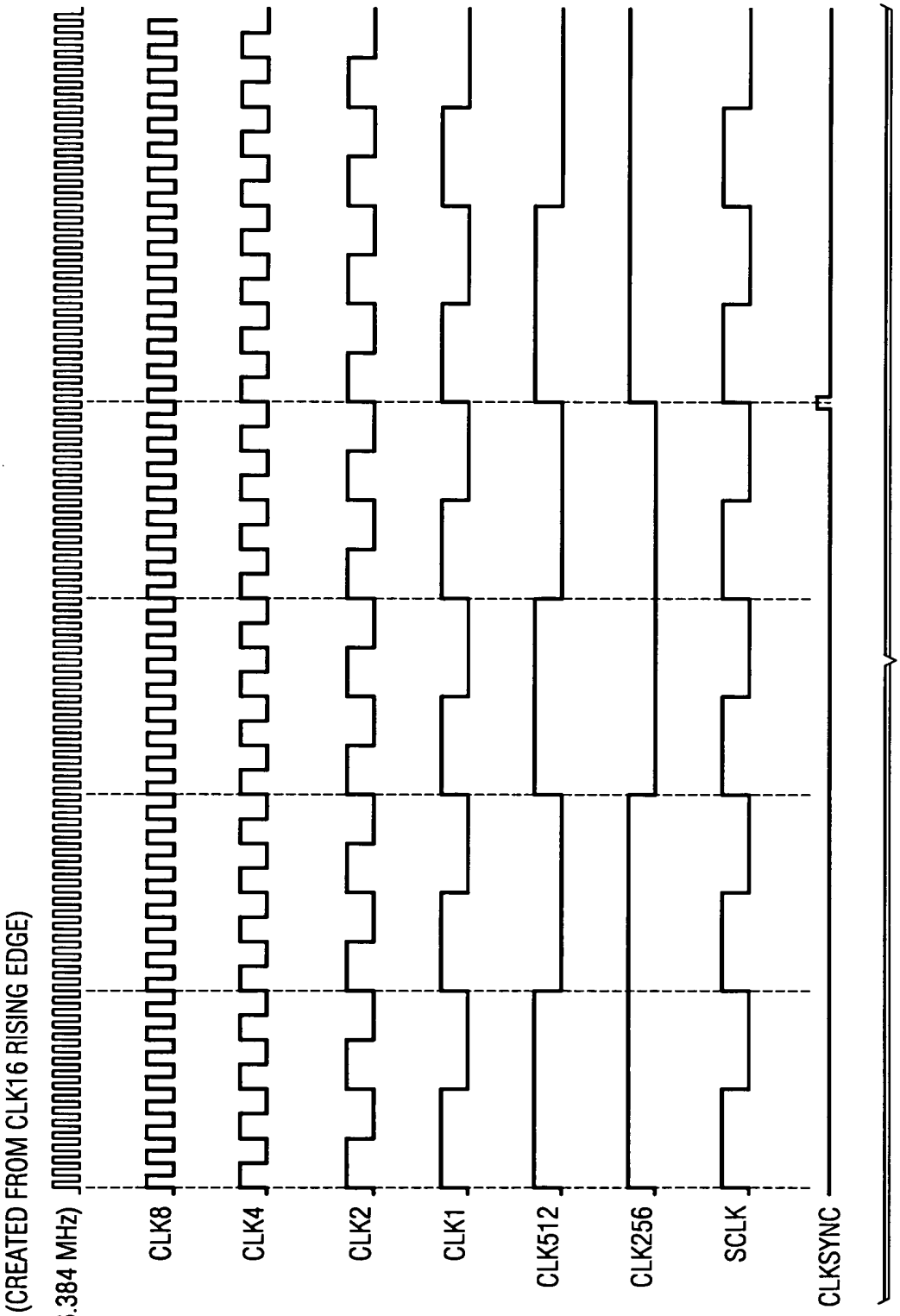


FIG. 36A

FIG. 36B